N° d'ordre : D.U : 2637
EDSPIC : 730

Université Blaise Pascal - Clermont II

École Doctorale
Sciences Pour L'Ingénieur De Clermont-Ferrand

Thèse

présentée par :

Claude Aynaud

pour obtenir le grade de

Docteur d'Université

Spécialité : Vision pour la robotique

Localisation précise et fiable de véhicules par approche multisensorielle

---

Préparée à l'Institut Pascal et
Soutenue publiquement le 8 décembre 2015 devant le jury :

M.	Abderrafiaa	Koukam	Prof. Univ. Tech. Belfort-Montbéliard	Président de jury
M.	Maan	El Badaoui El Najjar	Prof. Univ. Lille 1	Rapporteur
M.	Helder	Araújo	Prof. Univ. Coimbra	Rapporteur
M.	Christophe	Debain	Chargé de recherches, Irstea Clermont	Examinateur
M.	Romuald	Aufrère	Maître de conférences Univ. B. Pascal.	Examinateur
M.	Roland	Chapuis	Prof. Univ. B. Pascal.	Directeur de thèse

Ce travail a bénéficié d'une aide de l'État gérée par l'Agence Nationale de Recherche au titre du programme « Investissements d'avenir », d'une aide de l'Union Européenne (Fonds Européens de Développement Régional -FEDER- de la région Auvergne) et d'une aide de la Région Auvergne dans le cadre du LabEx IMobS3 (ANR-10-LABX-16-01).

Remerciements

En premier lieu, je tiens ici à remercier Roland Chapuis, Romuald Aufrère et Christophe Debain qui m'ont encadré et épaulé tout au long de cette thèse. Sans leurs conseils et leur soutien quotidien, cette thèse n'aurait certainement pas aussi bien avancé.

Je remercie mes rapporteurs Messieurs Helder Araujo et Maan El Badaoui El Najjar qui ont accepté de rapporter ma thèse. Je remercie également le président de jury Monsieur Abder Koukam.

Je remercie spécialement Coralie Bernay-Angeletti qui a commencé sa thèse peu de temps après moi et avec qui nous nous sommes mutuellement soutenus, surtout quand il s'agissait de ne pas craquer devant le calcul fastidieux des probabilités. Je remercie également Laurent Lequièvre qui m'a aidé pour toute la mise en place du SIG et qui a toujours été présent pour toutes mes questions concernant le développement.

Je tiens à remercier mes partenaires de jeux de société pour nos parties du midi, à savoir Clément Deymier, Morgan Slade, Ange Nizard, Stéphane Witzmann, Caroline Lemaitre.

Je remercie tous ceux qui ont partagé mon bureau à un moment ou à un autre, Guillaume Bresson, Thomas Féraud, Coralie Bernay-Angeletti, Laetitia Lamard, Mihai Chirca, Laurent Delobel, Alexis Wilhem, Mohammed Amine Boussadi, Satoshi Fujimoto.

Je remercie toutes les personnes que je n'ai pas citées, par manque de place ou par peur d'en oublier, qui m'ont apporté une aide dans mes travaux que ce soit par leur bonne humeur ou un apport technique.

Je remercie toute ma famille, en particulier mes parents pour leur soutien tout au long de cette thèse. Enfin et surtout, je tiens à remercier ma femme Anne-Elisabeth sans qui cette thèse n'aurait peut-être pas vu le jour, et ma fille Apolline, née le même jour que Romuald avec qui je conserverai ainsi toujours un lien.

Résumé

La localisation d'un véhicule est une étape cruciale dans le développement des véhicules intelligents. Les recherches sur ce sujet ont connu un grand essor ces dernières années. L'accent est souvent porté sur la précision de la localisation, nous présentons ici une méthode de localisation sur carte existante dont l'objectif est d'estimer la position du robot non seulement de façon précise mais également de manière fiable. Pour ce faire, l'algorithme développé se présente en deux étapes principales : une étape de sélection et de perception des informations les plus pertinentes et une étape de mise à jour de la position estimée et de la fiabilité, cette dernière étape permet également de détecter et réparer ou éliminer les précédentes erreurs.

La perception de l'environnement est réalisée à travers différents capteurs associés à des détecteurs spécifiques. L'humain utilise aussi différents capteurs pour se localiser et va intuitivement sélectionner le plus performant, s'il fait jour il utilisera ses yeux, sinon son oreille ou le toucher. Nous avons développé une approche similaire pour le robot qui tient compte des contraintes environnementales et de l'estimation actuelle pour sélectionner à chaque instant l'ensemble capteur, indice de la scène et détecteur le plus pertinent. La phase de perception, étant pilotée par un processus Top-Down, peut bénéficier d'informations déjà connues permettant ainsi de se focaliser sur l'indice recherché et d'améliorer les phases de détection et d'associations de données. Cette approche Top-Down s'appuie sur un réseau bayésien. Ce dernier permet de modéliser les interactions entre les différents événements qui se produisent en gérant l'incertitude. Il permet une prise en compte facile des différents événements. Par ailleurs le réseau bayésien présente une grande flexibilité pour l'ajout d'événements supplémentaires pouvant engendrer des non-détections (tels que dysfonctionnement de capteurs, conditions météorologiques, etc.).

Les données de l'environnement sont rendues disponibles grâce à une carte géoréférencée préalablement fournie. Avec le développement de cartes disponibles facilement sur internet, cette façon de faire permet d'exploiter au mieux l'information déjà fournie. L'utilisation d'une carte géoréférencée permet d'avoir un référentiel commun entre tous les véhicules ou éléments de l'infrastructure facilitant ainsi l'échange d'informations et ouvrant du coup la possibilité d'interaction simplifiées dans le cas de flottes par exemple.

Les résultats montrent que l'approche développée est pertinente pour une localisation précise et fiable aussi bien statique que dynamique. L'ajout de nouveaux capteurs se fait naturellement et sans nécessiter d'heuristique particulière. La localisation obtenue est suffisamment précise et fiable pour permettre des applications de conduite autonome utilisant, entre autres, cet algorithme.

Mots clefs : localisation, approche Top-Down, réseau bayésien, Système d'Information Géographique, multi-capteurs.

Abstract

Vehicle localization is a crucial step in the development of smart vehicles. The research in this domain has been growing in recent years. Generally, the effort is focused on the localization accuracy, we present here a localization method on existing map where the objective is to estimate the robot position not only with accuracy but also with confidence. To achieve this , the algorithm developed has two main steps : one, selection and perception of the most relevant informations and two, position estimation and confidence update. This last step also allows to detect and eliminate the previous errors.

Environment perception is well achieved, thanks to different sensors associated with specific detectors. Humans use different senses, shifting automatically in order to localize themselves depending on the situation of the environment, for e.g if there is enough illumination we depend on eyes, else the ear or the touch otherwise. We have developed a similar approach for the robot that takes into account the specific environmental constraints and actual position estimation to select at each instant the most relevant set of sensor, landmark and detector. The perception step, led by a top-down process, can use already known informations allowing a focus on the searched landmark and an improvement of the detection and data associations steps. This top-down approach is well implemented, thanks to a Bayesian network. Bayesian network allows to model the interactions between the different probable events with management of the uncertainty. With this network, it is very easy to take into account those different events. Moreover, a Bayesian network has a great flexibility to take into consideration additional events that can cause false detections (like sensor failure, meteorological conditions and others).

The environment data is obtained with a Georeferenced map (from GIS). With the already available maps on the internet, allows to exploit an already existing information. The use of a Georeferenced map facilitates the communication of informations between a vehicle and several other vehicles or with an element of the infrastructure, that can be very useful for multi vehicle coordination, for example.

The results shows that the developed approach is very accurate and reliable for localization, whether static or dynamic, and can be applied for autonomous driving. Moreover, new sensors can be added at ease.

Keywords : Localization, Top-Down approach, Bayesian network, Geographical Information System, Multi-sensors.

Table des matières

Table des figures

Liste des tableaux

CHAPITRE I

Introduction

Sommaire

I.1 La localisation et ses applications

« Terre, terre » crie la vigie du haut de la Pinta. Nous sommes le 12 octobre 1492, Christophe Colomb est convaincu d'être arrivé aux Indes après un voyage d'un peu plus de trois mois. En réalité, les navires sont en vue d'une île de l'archipel des Bahamas, San Salvador, soit une erreur d'estimation de la position de l'ordre de quelques milliers de kilomètres. . .

Dans l'Antiquité, les anciens avaient déjà conscience de l'importance d'une bonne localisation et ont cherché des outils et des méthodes dans ce sens. Différents outils de mesure ont été développés comme l'astrolabe (figure I.1a, instrument permettant de déterminer une latitude par la mesure de la hauteur méridienne du Soleil ou de sa distance zénithale), la boussole (figure I.1c utilisant le champ magnétique terrestre), l'octant puis le sextant (figure I.1b) permettant tous les deux la mesure d'angles, le sextant autorisant le calcul d'angle entre deux points par rapport à l'octant qui mesure l'angle d'un point par rapport à l'horizon) .

(a) Astrolabe

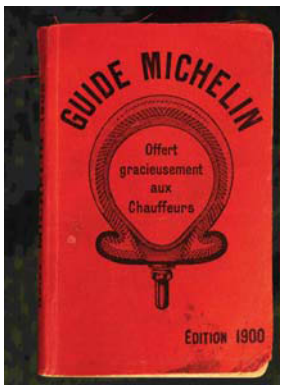
(b) Sextant

(c) Boussole

FIGURE I.1 – Différents instruments de mesure : un astrolabe, un sextant et une boussole

L'homme a également construit des points de repères lui permettant de se localiser par rapport à leur position. C'est ainsi par exemple qu'une des sept merveilles du monde a été bâtie entre -299 et -289 : le phare d'Alexandrie qui permettait aux navires de savoir où ils se trouvaient par rapport au port. Les lieux et leur localisation relative sont mémorisés sous la forme de cartes comme *l'itinéraire d'Antonin* (IIème siècle après Jésus-Christ), *la table de Peutinger* (IVème siècle), les cartes de Magellan (XVIème). Beaucoup d'efforts ont été déployés par l'humanité pour répondre à

(a) Carte Michelin au début du XXème siècle



(b) Guide Michelin au début du XXème siècle

FIGURE I.2 – Carte de France et guide Michelin

cette unique question : où suis-je ? La réponse à cette question est primordiale pour de nombreuses applications : la propriété, la guerre, le commerce, la correspondance épistolaire, les secours...

Avec le développement des véhicules motorisés, le besoin de localisation s'est fait ressentir d'une manière encore plus accrue. En 1910, la manufacture de pneumatiques Michelin commercialise pour la première fois des cartes de France (figure I.2a) et le guide touristique Michelin (figure I.2b) permettant aux particuliers de se repérer, de connaître les lieux de restauration, les hôtels, et donc de faciliter leur déplacement. Dans le même temps, la manufacture met en place sur le terrain les fameux panneaux en béton donnant les noms de villes ou villages et plus tard les directions, des informations sur la route et la conduite.

Ce qui est vrai pour les automobiles l'est aussi pour les avions. Le courrier est acheminé sur de longues distances par la voie des airs. Les pilotes doivent donc savoir se repérer dans les airs. C'est pourquoi dans les années 1920, l'Aéropostale installe en France les phares aéronautiques (figure I.3) qui permettaient aux pilotes effectuant un vol de nuit de connaître leur position et leur direction.

En 1957, le premier satellite artificiel Spoutnik I est lancé par l'URSS. Presque 40 ans plus tard, en 1995, le système de géolocalisation par satellites GPS développé par l'armée américaine est opérationnel. En 2000, ce système est accessible pour des applications relevant du domaine civil. Depuis, régulièrement de nouvelles méthodes, de nouveaux capteurs voient le jour avec comme objectif la localisation par rapport à un but fixé.

I.2 Application à la robotique

I.2.1 Présentation de la robotique

En parallèle des techniques de localisation, nous avons vu se développer le domaine de la robotique. Le mot « robot » a été créé en 1920 par l'écrivain tchèque Karel Çapek, dans une de ses pièces de théâtre (R. U. R. [Rossum's Universal Robots]), pour dénommer un androïde construit par un savant et capable d'accomplir tous les travaux normalement exécutés par un homme. Cependant le concept n'était pas nouveau et

FIGURE I.3 – Phare aéronautique de Montferrand construit en 1927

l'idée de robots avait déjà germé. Des automates avaient été conçus au le cours du premier siècle après Jésus-Christ par Héron d'Alexandrie. Les automates connaissent un essor important au XVIIIème siècle et vont continuer de se développer par la suite.

(a) Horloge astronomique de Lyon datant de 1379 : un automate décrivant le mouvement des étoiles, du soleil

(b) Automate capable de jouer de l'orgue (construit entre 1767 et 1774)

FIGURE I.4 – Exemples d'automates

Les robots s'inscrivent dans la suite logique des automates. La différence essentielle étant l'utilisation par le robot de capteurs lui permettant d'appréhender son environnement et réagir en conséquence tandis qu'un automate accomplit toujours la même série d'actions. Les robots ont rapidement su prouver leur utilité notamment dans l'accomplissement de tâches dangereuses, difficiles ou répétitives. L'industrie sera le premier secteur à utiliser des robots pour le travail à la chaîne, le chargement, etc. (figure I.5).

Le domaine militaire est également impacté avec le développement de drones, (fig-

(a) Chaîne de montage Citroën

(b) Robot KUKA chargeant des palettes

FIGURE I.5 – Exemples de robots industriels

ure I.7b, avions sans pilote permettant d'effectuer des missions de reconnaissance, de guidage, . . .), de véhicules terrestres autonomes pour le transport, la reconnaissance, la surveillance de sites sensibles, etc.

Les robots commencent à être plus présents parmi nous en s'intégrant directement chez les particuliers. Ces robots peuvent aider aux tâches domestiques comme le Roomba aspirant les sols (figure I.6a), à la mobilité des personnes âgées [48], voire pour des tâches comme l'éducation avec le robot Nao [61] (figure I.6b).

(a) Aspirateur autonome Roomba 780 mis au point par l'entreprise iRobot

(b) Robot Nao commercialisé par la société Aldebaran

FIGURE I.6 – Robots domestiques

Il est possible de distinguer deux grands types de robots, les robots fixes comme

les bras manipulateurs et les robots mobiles. Les robots mobiles peuvent également être décomposés en différentes catégories suivant le milieu où ils évoluent : aquatique (figure I.7a), terrestre, ou aérien (figure I.7b).

(a) Un robot aquatique : le Crabster

(b) Un robot aérien : le Drone Harfang

FIGURE I.7 – Robotique mobile : exemples de robots aquatique, aérien

I.2.2 La robotique mobile terrestre

Nous sommes arrivés dans une ère où beaucoup de tâches, ordinairement réalisées par l'homme, sont maintenant effectuées de manière automatique. Nous nous intéressons ici aux robots terrestres. Ils peuvent eux-mêmes se décliner en de nombreuses versions : le Roomba (figure I.6a) ou Nao (figure I.6b) sont des robots terrestres qui présentent de fortes différences au niveau du design, de l'application visée, du moyen de locomotion, ... La robotique terrestre peut impacter de nombreux domaines comme les tâches ménagères, l'agriculture, les voitures autonomes, etc. Nous nous focalisons sur les robots équipés de roues. Les applications peuvent aller de notre propre confort comme les systèmes d'aides à la conduite (par exemple le créneau automatique d'une voiture) jusqu'à des utilisations dans des environnements potentiellement dangereux pour l'homme (intervention dans une zone nucléaire, surveillance de l'activité volcanique, ...) en passant par les exploitations agricoles.

En octobre 2010, dans un but publicitaire, Google lance la Google Car (figure I.8a), voiture sans conducteur immatriculée dans l'état du Nevada et qui permet de se déplacer d'un point A à un point B sans intervention humaine.

(a) La Google Car

(b) Le poids lourd Freightliner Inspiration

FIGURE I.8 – Deux véhicules autonomes autorisés à rouler au Nevada

FIGURE I.9 – La F015, concept de voiture autonome de luxe présenté par Mercedes au CES2015

La sortie de cette voiture a eu un gros impact médiatique. Les constructeurs automobiles et les équipementiers se sont lancés dans la course et développent désormais leurs propres systèmes. En janvier 2015 au Consortium Electronic Show (CES), Mercedes présente sa propre voiture autonome, la F015, qui déambule dans les rues de San Francisco (figure I.9). Plus récemment, en mai 2015, le Nevada a autorisé un nouveau véhicule autonome à circuler, un poids lourd cette fois, le Freightliner Inspiration (figure I.8b) commercialisé par le groupe Daimler.

Que ce soit pour des applications comme la voiture autonome ou l'épandage d'engrais dans les champs, il existe une étape importante et qui ne peut être négligée, c'est la localisation du véhicule. Les véhicules concernés sont différents technologiquement notamment au niveau des capteurs disponibles, de la vitesse à laquelle ils évoluent, . . . , mais l'étape de localisation est incontournable pour envisager une conduite autonome. Les véhicules se déplacent dans des environnements complexes et variés. Il existe aujourd'hui de nombreuses technologies très performantes, la Google Car par exemple a parcouru des milliers de kilomètres et a eu seulement quelques accrochages sans gravité qui ne sont pas imputables directement à la partie logicielle. Cependant les capteurs utilisés sont souvent à un prix prohibitif interdisant, pour le moment, une application grand public. Nous préconisons alors un système capable de s'adapter aux capteurs présents sur la voiture et plus généralement capable de tirer le maximum de profit de l'information disponible pour localiser le véhicule.

I.3 Objectifs de la thèse

L'objectif de cette thèse est d'estimer de manière précise et fiable la position d'un robot mobile terrestre. Cette estimation se fera en exploitant au mieux les différentes informations disponibles (fournies par les capteurs, les détecteurs associés, une carte de l'environnement, . . .). Ces différentes informations seront utilisées de manière à tou-

jours tendre vers l'objectif de localisation que nous nous serons fixés. Cet objectif devra être défini avec soin. Nous verrons notamment qu'il aura en réalité deux composantes : la précision et la confiance dans l'estimation effectuée. Ces deux composantes sont indispensables dans des applications où le véhicule peut être commandé de manière automatique. Les objectifs de précision et de confiance sont fixés en fonction de l'application finale envisagée.

Le véhicule doit également être capable d'échanger des informations comme sa position avec d'autres véhicules, ou d'autres éléments du réseau comme un serveur distant par exemple. Une application envisageable est par exemple le déplacement en convoi, chaque véhicule aura alors besoin de connaître la position de ses congénères. Il ne pourra donc pas utiliser une localisation relative qui n'aura de sens que pour lui. Un référentiel commun doit être utilisé, la façon la plus simple et la plus générique est d'utiliser les coordonnées géographiques. Ce système de référencement permet d'une part de faciliter les communications entre les véhicules et d'autre part autorise l'utilisation de cartes géoréférencées déjà disponibles sur le marché.

Une contrainte est aussi donnée au niveau des capteurs. Les véhicules disponibles ne sont pas tous équipés de la même manière mais ils doivent cependant pouvoir tous utiliser le même algorithme. Il n'est pas envisageable dans notre cas de développer une méthode de localisation spécifique pour un seul type de véhicule avec des capteurs prédéterminés. La méthode développée doit être générique et pouvoir être utilisée avec tout type de capteur, que ce soit un télémètre laser, un GPS, une caméra ou autre. Chaque type de capteur apporte une information différente et sera donc plus ou moins utile en fonction du contexte. Cela nécessitera une stratégie de fusion (qui représente la principale contribution de ce travail de thèse) et de traitement de l'information se basant sur le contexte, l'objectif et les données disponibles afin d'optimiser le résultat. Ceci sera réalisé avec une approche Top-Down s'appuyant sur un formalisme bayésien.

I.4 Contributions

Les principales contributions de cette thèse sont les suivantes :

Stratégie de localisation par exploitation d'une carte géoréférencée pré-existante Aujourd'hui de plus en plus de cartes géoréférencées sont disponibles. Il semble alors judicieux de s'appuyer sur cette information sur l'environnement qui est déjà disponible avant toute action de perception de la part du robot. Il n'y a plus besoin de construire de carte au fur et à mesure, ce qui évite notamment les problèmes de dérive, de choix du repère, etc.

Estimation simultanée de la position du robot et de son intégrité A chaque instant, le système développé estime non seulement la position du robot avec une certaine incertitude mais également la confiance qu'il a dans cette estimation, c'est-à-dire la probabilité que le robot se trouve effectivement dans l'espace d'incertitude autour de la position estimée.

Optimisation de l'information disponible à l'aide d'une méthode Top-Down inspirée par le comportement humain À partir de l'information déjà connue, un critère de sélection des données les plus pertinentes est mis en place. Ce critère permet de prendre en compte l'objectif visé (en termes de précision et d'intégrité), le contexte et les données disponibles pour avoir un résultat optimal.

Modélisation des interactions entre les événements à l'aide d'un réseau Bayésien Savoir modéliser les causes et les conséquences d'un événement particulier est

une étape importante. Par exemple, savoir pourquoi un élément de l'environnement n'a pas pu être perçu est important pour la prise de décision. Afin de répondre à cette problématique, un réseau Bayésien modélisant les incertitudes, les probabilités des différents événements et les relations de cause à effets a été mis en place.

Mise en place d'un processus de détection et de réparation des erreurs Il peut arriver que le système de perception soit pris en défaut et fournisse des résultats erronés. S'appuyant sur le réseau Bayésien mis en place, un système d'identification des erreurs et de la cause des erreurs est mis en place. Le système est alors capable d'éliminer ou réparer ces erreurs.

La validation de ces contributions a été réalisée dans un premier temps à l'aide d'un simulateur réaliste puis dans un second temps en environnement réel venant ainsi confirmer la pertinence des choix effectués. Ces travaux ont donné lieu à plusieurs articles publiés dans des conférences nationales [1, 4] ou internationales [2, 3], ainsi qu'à un article de revue soumis et en attente de décision. Le principe de la méthode développée a également été transposé en interne pour des applications de reconnaissance d'objets dans une image [5].

I.5 Organisation du mémoire

Après une introduction sur le contexte applicatif de ces travaux de thèse, un état de l'art sur les principales méthodes de localisation existantes est présenté dans le chapitre 2. A l'issue de ce chapitre, nous voyons que les informations disponibles sont de plus en plus nombreuses et qu'il faut donc une stratégie d'extraction de la connaissance. Le chapitre 3 présente justement les différentes sources de connaissance disponibles pour un robot, les différentes façons d'aller rechercher l'information et de la traiter. Les conclusions de cette analyse nous amènent alors à conceptualiser nos objectifs et à concevoir un système adapté. Dans le chapitre 4, nous décrivons le processus mis en place pour prendre en compte nos objectifs, le contexte, les différents capteurs disponibles pour optimiser notre action de perception. Le chapitre 5 décrit la manière de modéliser au mieux l'environnement et les interactions entre les différents événements dans un contexte dynamique. Le chapitre 6 montre comment est effectuée la mise à jour de la position estimée et de la confiance dans cette estimation. Un système permettant de détecter les erreurs possibles et d'identifier leurs sources probables est présenté ainsi qu'une manière de les réparer. Le chapitre 7 présente l'implémentation de ces méthodes dans des environnements réels et simulés ainsi que les différents résultats. Les expérimentations sont effectuées dans des situations différentes qui permettent d'illustrer les divers aspects de la méthode. Les résultats sont alors analysés à l'aune des objectifs désirés.

CHAPITRE II

Localisation d'un robot mobile terrestre

Sommaire

II.1 Introduction

Pour accomplir une tâche précise, un robot doit être capable de connaître sa position avec précision et confiance. La précision désigne le degré d'incertitude métrique liée à la position estimée (par exemple, connaître la position du robot à plus ou moins 15 centimètres), la précision recherchée dépend de l'application visée. La confiance définit le degré de certitude d'être dans l'espace d'incertitude donné : si la confiance est nulle cela va signifier que le robot n'est probablement pas là où il pense être, si au contraire elle est égale à 1 alors le robot est là où il pense être, c'est-à-dire dans l'espace d'incertitude autour de la position estimée. Connaître une position signifie avoir un référentiel donné, un repère dans lequel se positionner. Rigoureusement, la position d'un robot est connue par sa position dans l'espace et par ses différentes orientations. Ces diverses orientations peuvent être caractérisées par exemple par les angles d'Euler, au nombre de trois, et également appelés roulis, tangage et lacet (voir figure II.1). Au total, il faudrait donc une localisation en 6 dimensions.

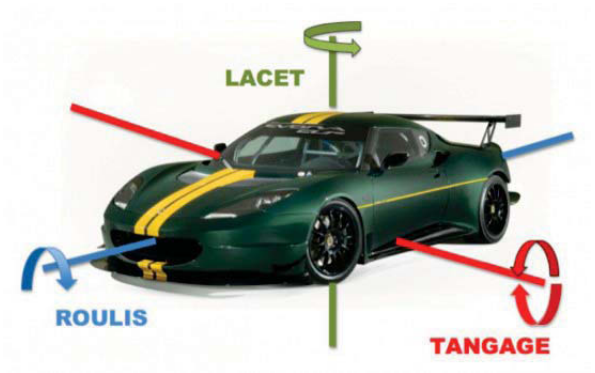


FIGURE II.1 – Définition du roulis, tangage et lacet

Cependant une approximation courante dans la robotique mobile terrestre est d'utiliser seulement deux composantes 2D de position (x, y) et une composante d'orientation (angle de lacet) du véhicule caractérisée par l'angle θ_R (voir figure II.2). Nous parlons dans ce cas de 2D $1/2$. Cette approximation qui simplifie grandement les calculs se révèle généralement suffisante pour localiser un véhicule terrestre destiné à des applications routières ou urbaines par exemple. Elle ne peut toutefois s'appliquer que dans un monde quasiment plat, son utilisation dans un terrain très accidenté ou montagneux ne serait pas judicieuse.

Nous pouvons distinguer deux catégories de localisation : la localisation locale et la localisation absolue. La localisation locale peut être la perception du mouvement relatif du robot, souvent réalisée à l'aide de capteurs proprioceptifs comme l'odomètre, un capteur d'angle de braquage, une centrale inertielle ou à l'aide de capteurs extéroceptifs avec des techniques comme l'odométrie visuelle [110]. Ces approches souffrent toujours d'un problème de dérive due à l'accumulation d'erreurs au cours du temps [33]. Généralement cette dérive inhérente ne rend cette méthode de localisation utile que pour des petites distances. Afin de se localiser sur de plus longues distances il devient nécessaire d'utiliser des données de l'environnement. Ces données vont être stockées dans des cartes. Une carte est un moyen simple, efficace et intuitif, utilisé par l'humain pour se repérer, décrire sa position à d'autres personnes, etc. Quelques études

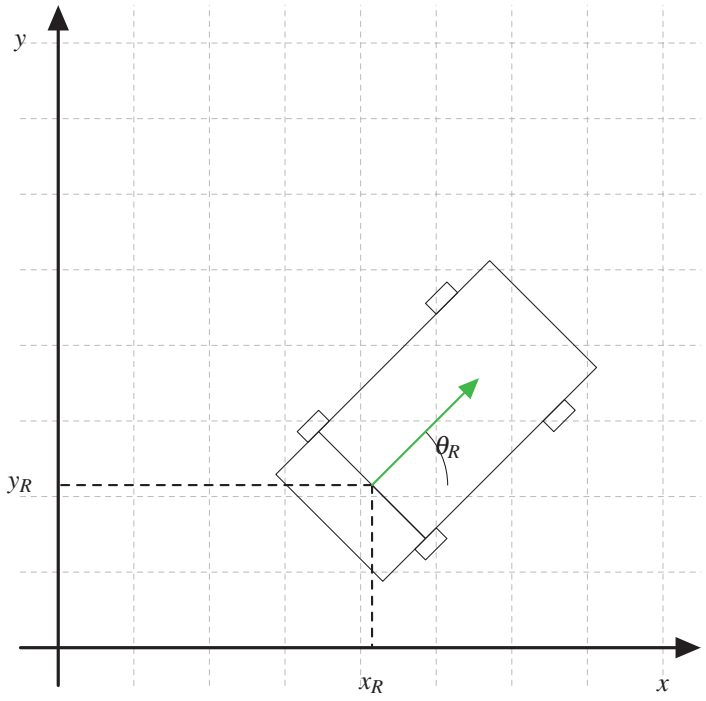


FIGURE II.2 – Représentation du robot dans un environnement en deux dimensions

ont prouvé que les animaux utilisaient également le concept de carte pour se déplacer et donc se localiser [137,146]. Les algorithmes de localisation et de navigation utilisent donc naturellement ces cartes pour aider le robot à se localiser et également, par exemple, pour planifier sa trajectoire. Toutefois, si la notion de carte est facile à saisir, les cartes sont nombreuses et variées, ne représentent pas les mêmes informations, n'ont pas la même précision, ne sont pas accessibles par les mêmes techniques, etc. Les systèmes de localisation et/ou navigation exploitant des données GPS peuvent également être considérés comme utilisant une carte : en effet la position des satellites est connue et l'ensemble de leurs positions peut être vu comme une carte, en l'occurrence une carte dynamique dans laquelle les objets bougent au cours du temps. La carte pourra également dans certains cas être directement fournie au robot, qui l'utilisera pour la localisation et, dans d'autres cas, le robot devra construire lui-même sa carte, lors d'une phase de cartographie, pour se localiser dedans dans le même temps : c'est ce qu'on appelle du SLAM pour Simultaneous Localization And Mapping. Dans cette thèse, nous partons du principe que la carte est déjà fournie au robot. S'il désire se documenter sur les stratégies de construction de cartes, le lecteur pourra se référer à [100] ainsi qu'à toutes les techniques de SLAM.

Pour étudier la localisation d'un robot dans une carte, nous verrons ceci en deux parties. Dans un premier temps, nous verrons les différents types de cartes qui existent puis, dans un second temps, les différentes méthodes qui ont été développées et qui s'appuient sur ces cartes.

II.2 Les cartes

Qu'est-ce qu'une carte ? D'après le petit Larousse 2014, une carte est une *Représentation conventionnelle, généralement plane, de phénomènes concrets ou même abstraits, mais toujours localisables dans l'espace*. Les cartes sont omniprésentes aujourd'hui et des entreprises ou des organismes d'état sont spécialisés dans leur création. Nous pensons notamment à l'IGN, TomTom, les cartes routières publiées par des entreprises comme Michelin, etc.

II.2.1 Représentation des informations cartographiées

Les données cartographiées sont généralement représentées sous forme métrique ou sous forme topologique. Les cartes topologiques stockent l'information relative entre les objets tandis que les cartes métriques utilisent des informations de position absolue.

Cartes topologiques

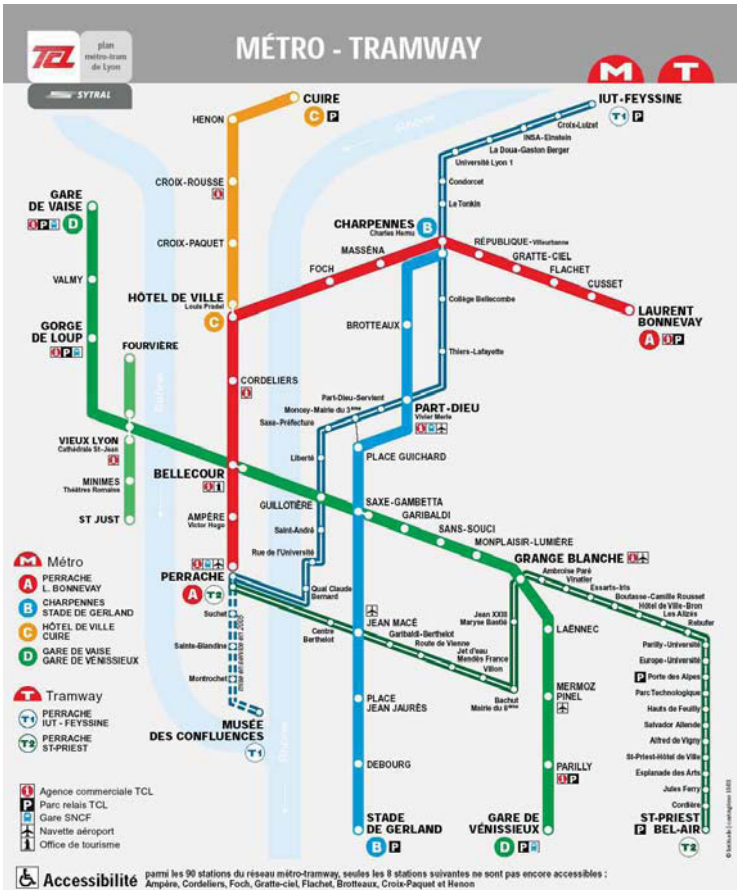


FIGURE II.3 – Un exemple de carte topologique : le plan de transports en commun de Lyon

Ces cartes comportent des nœuds et des liens. Chaque nœud va être relié à d'autres nœuds via les liens. Cette représentation se prête bien à une modélisation sous forme de graphe où les liens sont les arcs entre les nœuds. Les cartes routières ou les cartes sémantiques (cf II.2.2) peuvent également être représentées de manière topologique. Nous utilisons parfois naturellement ce type de représentation : « Le bureau A est le troisième bureau sur la droite depuis la porte d'entrée ». Ces cartes peuvent donc être très utiles par exemple pour de la planification de trajectoires et ainsi donner des ordres naturels au robot (« va au bureau A » plutôt que « va au point (x, y) »). Un exemple commun de carte topologique est un plan de métro (voir figure II.3) où les distances entre les stations ne sont pas forcément bien modélisées. Ce que l'utilisateur regarde en premier ce sont les connexions entre les différentes lignes ainsi que l'ordre des stations.

Cartes métriques

Dans une carte métrique, les informations stockées sont directement référencées par leur position dans l'espace. Chaque information est indépendante des autres. La modification d'un élément (par exemple sa suppression) n'entraîne pas une modification de toute la carte.

II.2.2 Types de cartes

Nous avons vu dans la partie précédente comment les informations peuvent être représentées dans une carte. Chaque carte présentée ici peut éventuellement se décliner sous ces deux aspects (métrique et topologique). Nous allons maintenant voir quels types d'information peuvent être stockés dans une carte.

Cartes routières

Quand on parle de carte, généralement la représentation qui vient le plus rapidement à l'esprit est le plan des routes. Depuis très longtemps les gens exploitent ce genre de carte et aujourd'hui avec le développement d'outils de navigation grand public à base de récepteur GPS, les cartes routières sont développées, affinées et mises à jour régulièrement. Ces cartes sont des cartes topologiques le plus souvent et peuvent également être métriques. Elles sont utilisées pour la planification de trajectoires ou encore pour la localisation d'un véhicule en mouvement dont on sait qu'il est sur la route, ce qui va être le cas des systèmes de navigation grand public à base de récepteur GPS. En plus des routes proprement dites, elles vont généralement comporter des informations subsidiaires comme le nom des rues, les bâtiments remarquables, le type de voie (autoroutes, nationales, etc.).

Grilles d'occupation

Une grille d'occupation est une carte dans laquelle chaque partie de l'espace est représentée comme une case pouvant être soit occupée soit libre. L'espace est donc discrétisé et représenté comme une grande grille, chaque case étant connue par sa position et son état (occupé ou libre) (voir figure II.4). Chaque case possède une probabilité d'être occupée ou non. Cette probabilité va être calculée et remise à jour au fil du temps en prenant en compte l'état des cases voisines, les détections successives, etc. Par construction ces cartes sont des cartes métriques. Les méthodes les plus connues utilisent des cartes de l'environnement en deux dimensions [95] même s'il existe des travaux

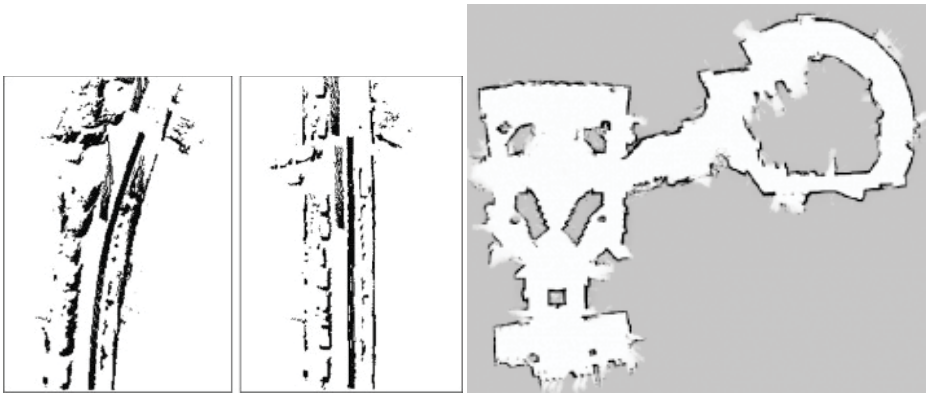


FIGURE II.4 – Représentation de l'espace sous forme d'une grille d'occupation

utilisant des cartes en trois dimensions [63, 64, 130]. Ces cartes tridimensionnelles permettent une meilleure modélisation de l'environnement mais sont plus coûteuses en terme d'occupation mémoire et de calcul. Cette façon de représenter l'environnement a été développée pour différents types de capteurs comme les capteurs à ultrasons, les télémètres laser, les caméras.

Cette représentation sous forme de grille est adaptable par essence à n'importe quel type de géométrie (ligne, courbe, polygone, etc.) puisque cette dernière n'est pas prise en compte. Toutefois, l'utilisation de ce type de carte va être fortement conditionnée par l'approximation locale de la fonction d'occupation directement engendrée par la discrétisation effectuée d'une part et par l'incertitude de l'estimation de la valeur d'autre part. Dans ce genre de cartes, par essence il ne peut y avoir d'incertitude sur la position des points mais en revanche, la principale source d'erreur portera sur l'état occupé ou libre d'une case. Ces erreurs peuvent provenir de plusieurs sources : soit d'un environnement dynamique dont la structure peut légèrement varier au cours du temps, soit d'une mauvaise identification lors de la phase de construction de la carte. Dans [144], une approche probabiliste a par exemple été développée pour pallier ce problème.

Les grilles d'occupation ne sont pas seulement des grilles de perception. Elles peuvent être des cartes de navigation et indiquer au robot quelles sont les zones navigables c'est-à-dire les zones où il peut évoluer [103].

Cartes géométriques

Nous appelons ici une carte géométrique, une carte dans laquelle sont représentées des formes géométriques comme des points ou des lignes par exemple [147]. Ces formes sont décrites ainsi que leur position dans l'espace. Dans les approches de type SLAM, un type de carte très populaire concerne les cartes de type point. Un point est une forme très simple, il n'a pas de dimension géométrique mais est souvent associé à un descripteur qui permettra de le détecter ultérieurement. Un point pourra par exemple être le résultat d'une détection de points de Harris. Un point peut sembler délicat à détecter, suivant l'environnement, l'angle sous lequel il est regardé et il pourra être sujet à du bruit. C'est pourquoi d'autres formes géométriques ont été développées comme des lignes droites, des lignes courbes ou même des surfaces [128]. Les environnements très structurés des villes ou de tout autre endroit façonné par la main de l'homme se prêtent volontiers à l'utilisation de formes géométriques de ce style. Nous avons parlé

d'un descripteur associé notamment pour les données de type point. En réalité ce descripteur est lui-même calculé pour des données spécifiques, par exemple des données caméra. Les cartes décrites ici stockent généralement des informations en fonction du type de capteur qui est utilisé tant pour la phase de cartographie que pour la phase de localisation.

Cartes sémantiques

Ces cartes introduisent la notion de sémantique et de classification des objets qui nous entourent. En effet, les cartes de types cartes géométriques ou grilles d'occupation ne se préoccupent guère de savoir quel est l'objet qu'elles représentent, seul leur importe le fait que cet objet soit visible ou non et quelle est sa forme. Une carte sémantique pour un robot mobile est une carte qui contient, en plus des informations spatiales de l'environnement, des données propres à l'objet représenté qui vont nous donner des informations sur ses fonctionnalités, son utilité, etc. Un objet ne sera plus pris sous un certain angle utile pour notre tâche (comme voir les murs seulement comme des lignes droites visibles avec un télémètre laser) mais dans sa globalité [112].

Les cartes sémantiques sont des représentations de plus haut niveau. Par exemple une ligne blanche au sol ne sera plus répertoriée comme une droite ou une courbe détectable par la caméra mais comme un passage piéton, une délimitation de route, etc. Ces différentes informations ne sont pas destinées à un capteur particulier et peuvent donc potentiellement être utilisées pour plusieurs capteurs. Par exemple si la carte indique un mur à tel emplacement et à telle hauteur, il est possible d'exploiter ces informations afin de le détecter avec un télémètre laser, de détecter les coins du mur avec une caméra, ou de penser que cela risque de gêner le GPS avec le phénomène de multi-trajets (voir figure II.5) , … Cette façon de représenter la carte est aussi plus proche de la façon dont l'humain se représente son environnement [146] et c'est l'une des raisons pour lesquelles le développement de cartes de ce type est intéressant en robotique mobile [148]. Ces cartes sont également développées dans le but de faciliter l'interaction humain/robot [104]. En revanche, leur construction est plus délicate que celle de cartes de type grille d'occupation ou de type géométrique [104] car elle implique une étape supplémentaire de classification des amers [1]. Certaines cartes sémantiques restent dédiées à des environnements spécifiques comme l'intérieur [21, 104] ou un environnement urbain [122, 134]. Dans [81], les auteurs dressent un panel des méthodes existantes pour la construction de cartes sémantiques pour des tâches dédiées à des robots mobiles.

Cartes absolues disponibles

Aujourd'hui, il y a de plus en plus de cartes disponibles et accessibles facilement. Une carte ne peut être partagée entre plusieurs personnes que si elles ont un référentiel commun. C'est pourquoi les cartes, qui sont accessibles, de manière large sont géoréférencées, simplifiant ainsi grandement leur partage, elles sont aussi dites absolues. Nous pouvons penser en particulier à des projets comme Map Quest, Open Street Map (figure IV.6), Google Maps. MapQuest est une société de cartographie assez ancienne fondée en 1967 qui fournit gratuitement des cartes en ligne. Depuis juillet 2010, MapQuest utilise les cartes d'Open Street Map. Open Street Map possède différentes caractéristiques intéressantes, notamment il est « Open » et fondé sur la col-

1. Un amer est un point de repère présent dans l'environnement, par exemple un poteau, un arbre ou un mur.

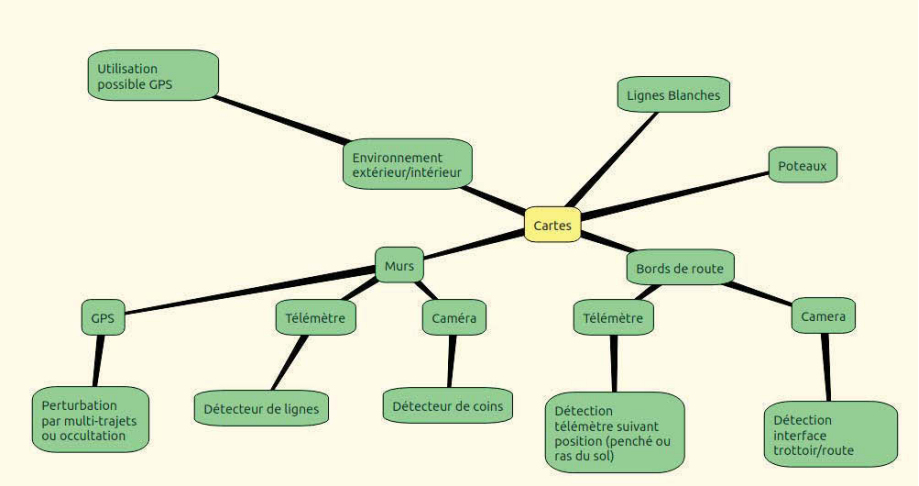


FIGURE II.5 – Utilisation d'une carte sémantique avec plusieurs types de capteurs : une même carte peut être exploitée par plusieurs types de capteurs, un mur pourra par exemple être détecté à l'aide d'un télémètre, une caméra ou encore nous donner des indications sur la fiabilité des données fournies par un récepteur GNSS

laboration de bénévoles. Les informations peuvent être extrêmement variées comme les routes, voies ferrées, les bâtiments, les voies de bus, les forêts, les courts de tennis, etc. Chaque personne peut rajouter des éléments, signaler une anomalie, un événement provisoire (travaux sur la route par exemple). Cette base peut être accessible via des requêtes comme pour une base de données classique. Malgré tout, Open Street Map comporte quelques inconvénients comme notamment la précision des données. En effet, la cartographie étant réalisée pour partie par des bénévoles, la qualité est assez hétérogène et dépendante du protocole suivi, de la qualité des capteurs, ... Pour les mêmes raisons, une confiance variable peut être accordée à chaque donnée. Cette confiance va dépendre principalement de la communauté qui a contribué à enrichir ces données. Malgré tout, il y a aujourd'hui de plus en plus de travaux s'appuyant sur Open Street Map et dans des domaines très variés comme la navigation sur routes, le SLAM [46], l'étude de l'évolution des rues [108]. Par ailleurs, l'utilisation de ces cartes peut être étendue. Par exemple, plutôt que de référencer les poteaux comme étant des cylindres d'une certaine hauteur et d'un certain diamètre, leur identification en tant que feux de signalisation ou poteaux d'éclairage public va permettre au système d'utiliser ceci comme information pour la navigation autonome. Ces cartes sont alors sémantiques.

Google Maps est un service de cartographie gratuit en ligne. Il est possible d'utiliser Google Maps à différents niveaux. Soit d'un niveau haut pour voir le dessin des routes, des principaux bâtiments, soit de zoomer et pouvoir voir des photos des endroits concernés et utiliser les données de Google Street View. Google Street View est un service permettant de visualiser un ensemble de photos prises par un véhicule géolocalisé préalablement passé par l'endroit concerné. Dans [98], les auteurs utilisent les données de Google Street View pour localiser un drone muni d'une caméra.

II.3 Les Techniques de localisation basées sur des cartes

Dans la partie précédente, nous avons vu plusieurs types de représentations de l'environnement dans lequel le robot évolue. Maintenant, nous allons voir comment le robot va se localiser dans ce même environnement notamment en se basant sur des cartes. Nous étudierons dans la suite plusieurs grandes méthodes de localisation. En présentant ces différentes méthodes de localisation, il n'est pas nécessaire de se préoccuper de savoir si la carte est locale ou absolue, en effet les techniques sont généralement identiques. Une exception existe cependant pour les systèmes GNSS qui sont par essence des capteurs permettant une localisation absolue dans leur utilisation classique.

II.3.1 La localisation par satellites

Incontestablement, le capteur le plus connu pour la localisation absolue est le récepteur GPS. Aujourd'hui ce capteur est totalement démocratisé avec un large spectre d'applications grand public notamment pour la navigation ou pour d'autres applications comme le mouvement des plaques tectoniques, le suivi du mouvement migratoire de certaines espèces... Le GPS est le résultat d'un projet américain lancé dans les années 60 et initialement dédié exclusivement à des applications militaires. Le GPS fait partie des systèmes de positionnement par satellites appelé en anglais GNSS (Global Navigation Satellite System). Le GPS n'est pas directement basé sur une carte mais nous pouvons considérer qu'il s'appuie sur des amers actifs dont la position est connue à chaque instant ce qui peut être considéré comme une carte. Ces amers sont les satellites. La méthode consiste à calculer la distance qui sépare le capteur de plusieurs satellites (figure II.6).

FIGURE II.6 – Localisation GPS

D'autres systèmes fonctionnent sur ce principe, on citera notamment le système russe GLONASS ou le système chinois Beidou. Un système européen nommé Galileo est également en cours de réalisation.

Les systèmes GNSS sont faciles d'utilisation, et utilisables par tous mais présentent quelques inconvénients : le multi-trajet spécialement dans les zones urbaines, les occultations (voir figure II.7), les interférences, les erreurs d'horloges, les erreurs atmosphériques, les erreurs du système de corrélation interne, etc.

Pour le système GPS classique par exemple, la précision de la position fournie par le récepteur est généralement entre 1 et 10 mètres. Cette précision est souvent insuffisante notamment si le but est de faire du guidage terrestre ou d'autres tâches de

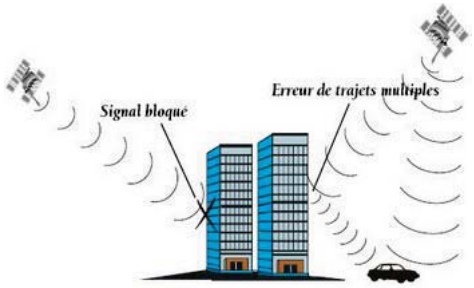


FIGURE II.7 – Différentes sources d'erreurs GPS

manière automatique. Il existe des solutions beaucoup plus précises comme le DGPS (Differential Global Positionning System) et le RTK (Real Time Kinematic). En sus des signaux émis par les satellites, le DGPS utilise des stations au sol. Ces stations sont fixes et connaissent leur position géoréférencée de manière certaine, elles peuvent ainsi servir de référence en transmettant les corrections à effectuer sur les pseudo-distances (une « pseudo-distance » est une mesure indirecte de distance par la mesure de l'instant de réception d'un signal daté à l'émission, lorsque les horloges de l'émetteur et du récepteur ne sont pas synchronisées). Les récepteurs GPS utilisant le RTK fonctionnent également sur le principe de stations au sol mais se basent sur la mesure du déphasage entre le signal reçu et un signal de référence. Ces deux types d'amélioration permettent d'augmenter substantiellement la qualité de l'estimation mais elles nécessitent d'équiper l'environnement, n'autorisant ainsi une utilisation que dans un contexte parfaitement contrôlé, et coûtent plus cher. Leur champ d'application reste également cantonné à des espaces suffisamment dégagés. Au final, ce système seul ne peut convenir pour des applications qui requièrent une localisation avec une grande disponibilité. Les systèmes GNSS vont ainsi souvent intégrer d'autres sources d'informations comme d'autres capteurs. Une source possible est l'utilisation de cartes. Par exemple, dans [90] les auteurs utilisent une carte de niveau pour affiner le résultat du GPS.

II.3.2 Exploitation des données fournies par la carte

Road-matching

L'objectif des méthodes de road-matching est d'identifier la route sur laquelle le véhicule se déplace. Ces méthodes sont basées sur les cartes routières. Pour ce faire, le véhicule dispose d'un récepteur GPS, souvent d'un odomètre et parfaois d'un gyromètre et d'un magnétomètre, et cherche ainsi à contraindre la position donnée par le récepteur avec le réseau routier en se basant sur l'hypothèse que le véhicule est sur la route (figure II.8). Cela exclut donc les zones non décrites dans la carte (parkings, champs, ...). C'est le principe utilisé dans les systèmes à base de GPS des automobilistes.

Dans [135], trois grands types de road-matching sont distingués : les approches géométriques, les approches topologiques et les approches avancées.

Dans les approches géométriques, chaque position estimée est reliée au point le plus proche de la carte [153] ou au segment le plus proche [119]. D'autres approches plus fines exploitent les résultats précédents et vont ainsi chercher à faire correspondre plusieurs estimations successives avec un arc de la route [153]. Cette approche est très

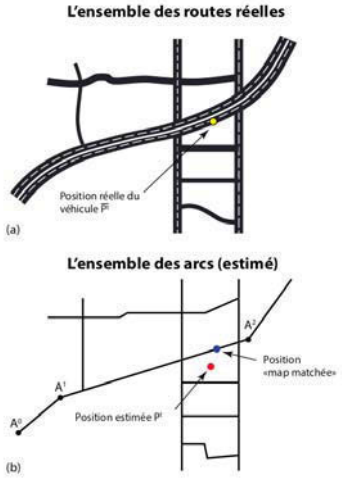


FIGURE II.8 – Méthode du map-matching (image tirée de [102])

sensible à l'initialisation [119].

Dans les approches topologiques, les caractéristiques de la route sont directement utilisées et reliées à la trajectoire du véhicule. Par exemple, [51] utilise l'orientation du véhicule et celle de la route (comparaison entre le segment reliant deux points GPS et les segments de route), la proximité de la position estimée par rapport au segment afin de savoir où le véhicule se trouve sur ce dernier. Malgré ces améliorations, ce type d'approche est souvent mis en difficulté par exemple à la sortie d'un carrefour [51].

Les approches avancées désignent les approches utilisant des filtres de Kalman, la théorie de Dempster-Shafer, la logique floue ou encore des filtres à particules.

Ces approches peuvent également intégrer la détection de données de l'environnement pour se localiser. Par exemple, [138] se localise sur une route en utilisant également la détection des lignes blanches au sol dont la position lui est fournie par la carte.

Map-matching

L'idée maîtresse du map-matching est de réaliser une carte locale de l'environnement autour du robot et de comparer cette carte locale avec une carte globale [45].

Une première technique est d'essayer d'appairer l'image capteur courante avec une image de référence [92]. Dans ce cas, la carte est en fait composée de différentes images capteurs prises à des endroits de référence. Le robot compare ses données capteurs avec la carte. Une méthode simple consiste à prendre la position des données correspondantes les plus proches [85], c'est-à-dire de chercher de quelle image de référence l'image capteur actuelle est la plus proche et de prendre comme position celle associée à cette image de référence. Cette méthode possède l'avantage d'être rapide mais ne pourra être précise que si la carte est très dense, nécessitant ainsi une construction de la carte fastidieuse et coûteuse en terme de stockage. Une méthode très populaire est l'Iterative Closest Point (ICP) [18]. Le principe de l'ICP est de minimiser de façon itérative la distance entre les points de l'image de référence et les points de l'image courante puis de calculer la transformation rigide permettant de passer de l'une

à l'autre. La principale difficulté de cette méthode est l'association de données, cette dernière sera souvent grandement conditionnée à l'initialisation. Il existe des méthodes plus robustes mais qui requièrent une puissance de calcul plus importante et peuvent nécessiter des techniques de programmation comme la parallélisation sur GPU [113]. Cette méthode peut être très rapide mais en contrepartie nécessite un échantillonnage important. Si l'environnement d'évolution est grand, l'espace de stockage nécessaire devra être conséquent sans compter que tout l'environnement devra lui-même être parcouru une première fois pour la construction de la carte de référence. Le point de vue sous lequel les données de référence auront été stockées est déterminant. Les conditions d'acquisition lors de la phase de cartographie sont elles aussi primordiales. Si ces conditions sont différentes lors de la phase de localisation, cela peut avoir un fort impact, comme la présence de soleil dans l'axe de la caméra. En résumé, cette méthode est assez simple, facile à mettre en place mais est assez contraignante sur les contraintes matérielles, l'orientation du véhicule, etc.

Une autre technique utilise les données capteur pour réaliser une grille d'occupation [127]. Cette grille d'occupation locale est alors comparée à une grille d'occupation globale. Le principal souci est de réaliser de mauvaises associations. Dans [20], un système multi-modal basé sur un RANSAC modifié permet de faire cette association dans des environnements de grande taille pour un résultat de 97% d'associations réalisées avec succès.

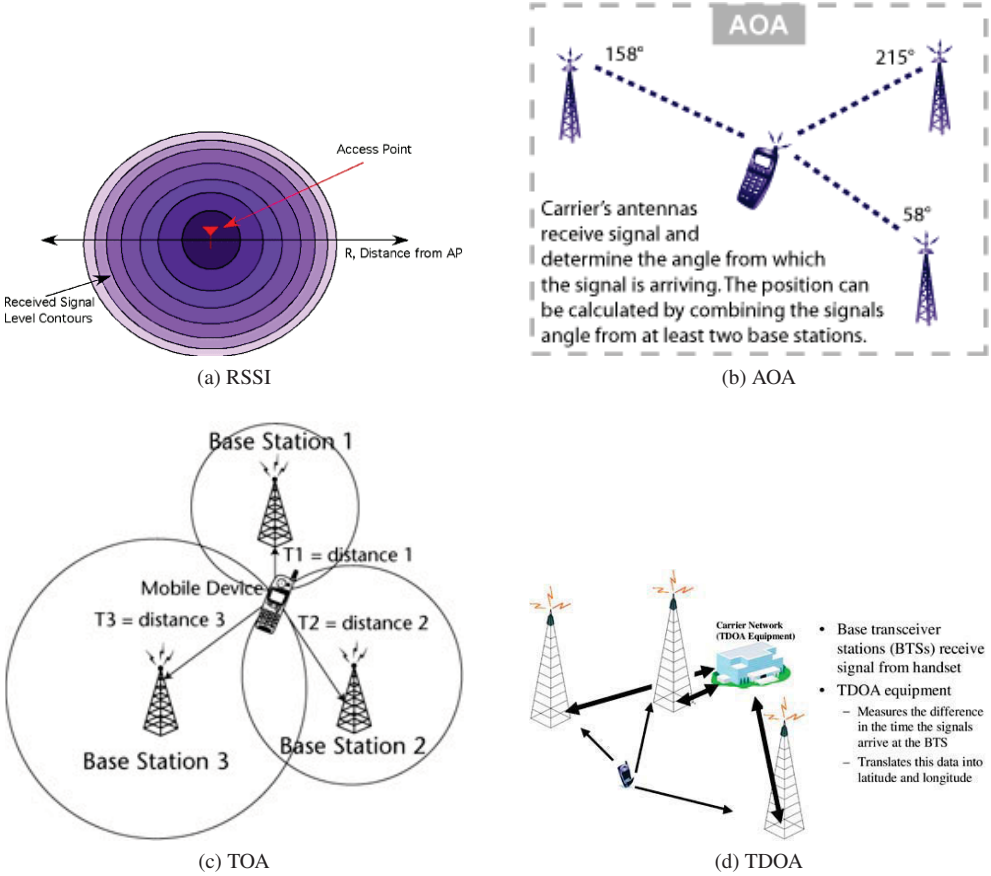
Une autre façon de faire est d'extraire des primitives depuis l'image capteur et de pouvoir ainsi comparer avec une carte géométrique. Par exemple, dans [80] les configurations des murs sont détectées à l'aide d'un sonar dans huit directions autour du robot. Cette carte des configurations est alors mise en correspondance avec une carte géométrique topologique. On retrouve ici le problème d'association précédent.

De ces différentes techniques de map-matching, nous pouvons tirer plusieurs principes généraux. Plus l'espace de recherche sera grand, plus la probabilité d'avoir plusieurs minima locaux sera grande. Pour pallier ceci, beaucoup de méthodes utilisent l'estimation de la position courante afin de réaliser la meilleure association possible. Les performances de ces méthodes dépendent grandement de la qualité des associations réalisées, si beaucoup d'amers sont utilisés le résultat peut être très performant mais requiert un fort coût calculatoire. Généralement les efforts seront donc portés sur les associations possibles et tout sera mis en œuvre pour qu'elles soient correctes. L'ensemble de l'image sensorielle étant utilisé à chaque fois, cela implique un niveau de bruit qui peut être fort (présence de piétons, aveuglement de la caméra, ...), il faut également des détecteurs suffisamment robustes.

Perception de données fournies par la carte

Une autre façon d'utiliser les données de la carte est de les considérer non plus de façon globale dans une scène mais de façon locale. Pour se localiser, le robot utilise des amers ou repères de l'environnement dont la position est connue. Le robot va donc les détecter, se positionner par rapport à eux et en déduire sa propre position. La carte va fournir la position des différents amers que le véhicule peut détecter et identifier grâce à ses capteurs. La première étape de ce type de processus va être la détection d'amers. Cette détection va dépendre de la nature de l'amer et des capteurs disponibles. Deux grands types d'amers existent : les amers actifs et les amers passifs. Les amers actifs sont des amers artificiels qui sont capables d'émettre un signal, par exemple infrarouge [82] ou Wifi [43], pour indiquer leur position. Plusieurs algorithmes ont été développés pour estimer la position du robot :

– indicateur de force du signal (RSSI) [54] (La puissance du signal est utilisée pour déterminer la distance entre l'émetteur et le récepteur) ;
– angle d'arrivée (AOA) [115] (Le système détermine l'angle entre l'émetteur et le récepteur) ;
– temps d'arrivée (TOA) [28] (la distance entre l'émetteur et le récepteur est déterminée par le temps mis par le signal) ;
– différence de temps d'arrivée (TDOA) [55] (une balise fixe et connue est utilisée comme référence) ;
– des méthodes hybrides [29].



(a) RSSI (b) AOA

(c) TOA (d) TDOA

FIGURE II.9 – Utilisation d'amers actifs, de gauche à droite et de bas en haut : RSSI, AOA, TOA, TDOA

Les amers passifs sont présents dans l'environnement et sont détectables grâce aux capteurs classiques comme la caméra, le lidar [129], ou un radar [152] par exemple. Les amers passifs peuvent être rajoutés artificiellement dans l'environnement, par exemple un ligne blanche tracée au sol que le véhicule doit suivre, ou bien des éléments déjà présents (le mur d'un immeuble par exemple). Par exemple, un arbre sera un amer passif détectable par la caméra ou encore par un lidar. Les murs, les trottoirs, les lignes de marquage au sol préexistantes sont également des amers passifs. Il est aisé de comprendre que l'utilisation d'amers passifs va dépendre en grande partie de l'environnement.

La localisation ne sera pas possible dans des environnements dépourvus d'amers détectables, elle sera également moins efficace dans des environnements fortement bruités avec la présence d'amers non cartographiés, d'amers cartographiés mais réellement absents, de neige, etc. La détection étant moins efficace, le problème se posant alors va être l'association entre les données détectées et les amers présents dans la carte. L'association de données est un problème récurrent et à l'origine de nombreux travaux [35]. Typiquement l'association de données se fait en deux étapes : un test qui détermine la compatibilité entre une observation capteur et un amer, et un critère de sélection qui permet de choisir la meilleure association possible. La méthode la plus simple et la plus intuitive d'association de données est l'association avec le plus proche voisin. L'idée maîtresse du plus proche voisin consiste à calculer la distance de tous les points obtenus avec le point recherché et de sélectionner parmi ceux restants celui dont la distance est la plus courte. La distance la plus souvent utilisée est la distance de Mahalanobis. Cette méthode est cependant loin d'exclure les mauvaises associations.

Afin d'éliminer certains problèmes d'associations de données, plusieurs méthodes ont été développées. [109] propose une approche intéressante : l'environnement, dans lequel le robot se déplace, a déjà été parcouru au préalable durant une phase d'apprentissage. Au cours de cette phase, les différents capteurs utilisés ont été évalués ; en effet, suivant l'environnement, les capteurs sont plus ou moins performants (une caméra est inutile dans un environnement de couleur uniforme et sans contraste, un GPS est inutile dans un environnement intérieur, etc.). Lors de la phase de rejeu, le système va alors sélectionner les capteurs les plus performants dans cette situation précise. La phase de détection, et donc d'association, sera donc plus performante. Malgré tout, cette technique impose un apprentissage préalable de l'environnement par le robot et n'utilise les différents capteurs que dans un ordre bien précis. Cela ne lui permet pas d'être réactif par exemple à un changement de condition.

D'autres techniques sont alors apparues pour répondre à ce problème d'association de données. Des approches ont été mises en œuvre comme le test de compatibilité jointe décrit dans [107] qui prend en compte le phénomène de corrélation des erreurs de prédiction. Des méthodes probabilistes assignant à chaque association possible une probabilité d'être correcte ont également été développées et sont connues sous le nom de JPDA (Joint Probabilistic Data Association) [13]. Elles sont régulièrement exploitées et améliorées [74, 136]. Une autre façon de faire consiste à ne plus faire une seule association mais à réaliser et propager toutes les hypothèses d'association possibles [145]. Cela revient à gérer plusieurs estimations de position possibles du véhicule au même instant. Au fur et à mesure que de nouvelles données arrivent, il est possible d'éliminer des hypothèses fausses ou au contraire mettre en avant des hypothèses qui étaient peu probables mais qui se révèlent justes. Le principal inconvénient de ce type d'approche est la propagation exponentielle des hypothèses impliquant un temps de calcul important et un stockage en mémoire non négligeable. Cette méthode est également délicate à mettre en place dans les systèmes où la carte est apprise ou mise à jour durant le déplacement du robot.

Impact de la précision des cartes

Dans toutes les méthodes présentées ci-dessus, une importante source d'erreurs vient de l'imprécision de la carte elle-même. Cette imprécision peut venir des défauts des capteurs utilisés lors de sa construction, des conditions extérieures (neige, pluie, aveuglement de la caméra par le soleil, ...) venant perturber les capteurs, de changements survenus dans l'environnement (construction de nouveaux bâtiments, de-

struction d'anciens, ...), etc. Les cartes peuvent également être de qualité très inégale suivant les lieux : cela est typiquement le cas pour Open Street Map [94]. Deux solutions peuvent être proposées pour remédier à ce problème : la construction de cartes beaucoup plus précises ou la gestion de l'imprécision par l'algorithme lui-même. La construction de cartes beaucoup plus précises est une opération coûteuse en temps et en ressources. En outre l'environnement étant souvent dynamique une carte décrit de moins en moins bien l'environnement avec le temps à moins de bénéficier de mises à jour régulières. L'autre méthode consiste à prendre en compte l'imprécision des cartes dans l'algorithme lui-même (qui pourra ounon réaliser la mise à jour de la carte). Par exemple, [14] développe une approche intéressante en se basant sur des cartes d'intérieur tracées à la main et donc nécessairement très imprécises. Les résultats montrent que le robot arrive à se localiser correctement dans plus de 80% des cas étudiés et est capable de suivre une trajectoire tracée à la main sur la carte fournie.

II.3.3 Le Cloud Robotics

Les algorithmes utilisés pour la localisation peuvent être gourmands en ressources et trop coûteux en performances pour être déployés sur tous les robots. Une réponse possible à ce genre de problématique peut être, au moins partiellement, résolue avec l'utilisation de serveurs distants.

Avec le développement du « Cloud » (exploitation de serveurs distants), les applications potentielles deviennent de plus en plus nombreuses et la robotique commence à être impactée ; on parlera ici de « Cloud Robotics ». Ce dernier possède de nombreux atouts [50, 75] : l'accès à beaucoup de données (datasets, trajectoires, cartes, librairies en accès libre etc.), l'accès à des grilles de calcul, partage d'information avec d'autres robots ou l'infrastructure, ... Le champ des possibles est donc largement étendu pour le robot. A titre d'exemple, le robot pourrait utiliser les services de « Google Goggles » [7], un service de reconnaissance d'images libre fourni par Google. Son fonctionnement est le suivant : l'utilisateur envoie une photo au service, ce dernier l'analyse et renvoie toutes les informations disponibles. Par exemple si le robot voit la Tour Eiffel et l'analyse avec ce service, celui-ci lui donnera toutes les informations sur la Tour et notamment sa localisation. Le robot pourra bénéficier d'une carte étendue dont la taille et la précision croissent de jour en jour.

Le Cloud Robotics va également accroître les possibilités de calcul du robot. Généralement, le robot supporte toute la charge de calcul nécessaire à sa localisation. Avec le Cloud Robotics, il est possible de déléguer cette charge à des grilles de calcul. Des méthodes qui jusque là semblaient performantes mais beaucoup trop lentes pour être mises en œuvre peuvent à nouveau être envisagées. Malgré tout, il peut être délicat d'envisager du temps-réel car les performances seront très dépendantes de la qualité de la communication dans le réseau. Cette partie fait l'objet de nombreuses recherches visant à assurer une utilisation temps-réel [16, 126].

L'échange d'informations entre les différents robots peut également s'en trouver simplifié. Chaque robot transfère ses données dans le Cloud et peut de la même façon accéder aux données transférées par les autres robots. Cela résout le problème du stockage de données, toutes les données sont stockées et datées à distance et accessibles à tout instant. Les capacités du robot peuvent donc être limitées, à titre d'exemple [118] montre un robot équipé seulement de capteurs et d'un pico-ordinateur de type raspberry Pi fonctionnant sur ce modèle.

Ces différents atouts sont attractifs, des architectures spécifiques commencent à voir le jour [101] ainsi que des applications de localisation et/ou navigation s'appuyant

sur cet outil [10, 15]. La quantité d'informations disponibles grâce au Cloud Robotics dépasse largement les applications actuelles et va donc imposer de nouvelles façons de gérer ces informations. Malgré tout, l'obligation d'avoir accès au réseau et d'être totalement dépendant de ce dernier freine son développement pour l'instant. Une solution élégante pour réduire l'impact de ce problème est suggérée dans [75] : les données seraient transmises au Cloud Robotics mais la plateforme qui enverrait les données les traiterait aussi en parallèle, au final les meilleurs résultats parvenus au bout d'un temps prédéfini seraient utilisés.

II.4 Conclusion

Les principaux problèmes rencontrés dans les diverses méthodes présentées concernent principalement l'association de données. Ce problème provient principalement de la présence de minima locaux, de bruit dans les données, de données corrompues. Ce problème peut parfois être minimisé au prix d'une forte puissance de calcul disponible qui permet la prise en compte de plus d'informations ou une meilleure exploitation des données déjà existantes. L'équipement de l'environnement permet aussi de pallier ce genre de défauts mais ne peut pas être applicable partout pour des raisons de coût lié à la mise en place et l'entretien de l'infrastructure. Une méthode efficace devra donc permettre de limiter au maximum les problèmes d'association en utilisant l'information disponible de manière efficace et ensuite d'être capable de détecter ces problèmes et les résoudre.

Dans les sections précédentes, nous avons pu voir que la représentation de l'environnement par une carte était loin d'être unifiée. Suivant les besoins, l'environnement était représenté de manière différente et d'un point de vue différent à chaque fois. Pourtant, l'environnement est rigoureusement le même pour tous, l'application finale qui est la localisation l'est également. De même, les techniques de localisation basées sur les cartes sont variées et pourraient l'être seulement en raison de l'hétérogénéité des capteurs utilisés mais ce n'est pas, loin s'en faut, la raison principale ; le nombre de capteurs utilisés pour la localisation est tout de même relativement réduit dans la pratique et la grande majorité des applications utilise les mêmes capteurs. Malgré cela, la diversité des techniques est importante mais elles sont toutes centrées autour d'un point important : l'utilisation des données fournies par les capteurs, la carte ou autres pour calculer la localisation d'un robot mobile. Elles peuvent toutes être résumées de manière globale de la façon suivante : « *Utiliser l'information disponible pour en extraire la connaissance relative au positionnement d'un robot mobile* ». L'avènement de techniques comme le Cloud Robotics où le nombre de données à gérer devient de plus en plus important rend cela encore plus crucial. C'est ce qui nous amène ici à considérer la localisation comme un problème de connaissance, ce que nous détaillons dans le chapitre suivant.

CHAPITRE III

Connaissance pour la localisation

Sommaire

III.1 Introduction

Les grands personnages « mettent à profit tout ce qu'ils voient, tout ce qu'ils entendent, ne négligent rien pour acquérir de nouvelles connaissances et tous les secours qui peuvent les conduire heureusement à leur fin », Sun Tzu dans l'Art de la Guerre.

Au chapitre précédent, nous avons pu voir différentes méthodes de localisation sur carte qui ont été développées pour les systèmes de robotique mobile. Ces systèmes sont nombreux et peuvent être très différents les uns des autres suivant le type de formalisme utilisé, les capteurs disponibles, les a priori (navigation sur routes, en environnement intérieur, ...), l'environnement d'évolution des robots (urbain, rural, ...), etc. Tout ceci est vaste et un formalisme unificateur n'est pas facile à mettre en place. Ce qui relie toutes les méthodes présentées est leur but qui est d'estimer la position du robot, c'est-à-dire de connaître, d'avoir la connaissance de cette localisation. Pour cela, elles essaient de gérer au mieux toutes les informations fournies pour en tirer de la connaissance au sujet de la localisation. Le problème de la localisation peut donc être considéré comme un sous-ensemble d'un domaine plus large : la connaissance, le sujet de cette connaissance étant la localisation. La connaissance dans sa globalité a occupé l'esprit des plus grands philosophes depuis longtemps. Le processus de production de la connaissance peut être formalisé suivant le schéma de la figure III.1 [22].

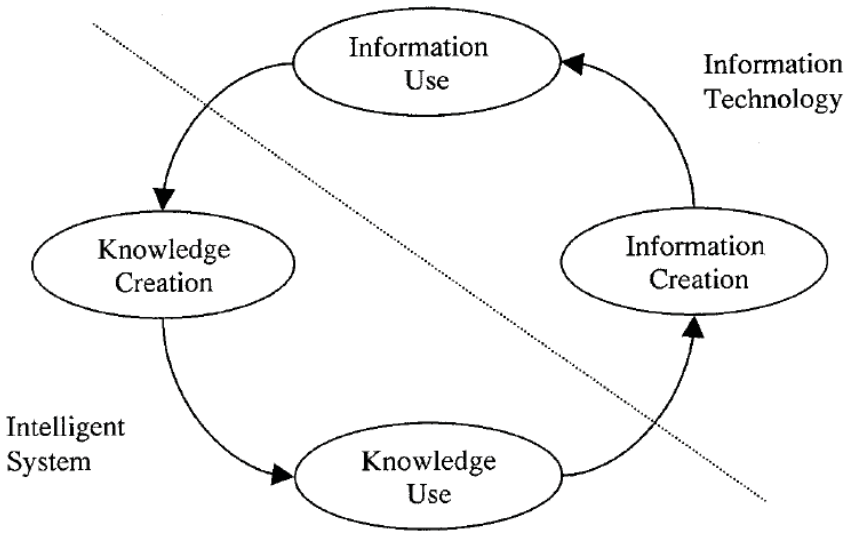


FIGURE III.1 – Cycle de production de la connaissance tiré de [22]

La connaissance est issue du traitement de l'information. Cette connaissance permet d'effectuer une action (par exemple la localisation peut être utilisée pour du guidage automatique), cette action va entraîner elle-même la création d'information (par exemple, le mouvement du robot suite à une commande va engendrer de nouvelles données capteurs), la boucle est bouclée. Dans le cadre de cette thèse, l'étape d'utilisation de la connaissance ne sera étudiée que succinctement à travers des exemples d'applications.

Nous nous intéressons ici à une infime partie de la connaissance, à savoir la connaissance de notre position.

Pour ce faire, nous considérerons plusieurs grandes parties :

– les sources de connaissance et leur contexte ;
– la façon d'accéder aux informations ;
– la gestion des informations pour en extraire la connaissance.

Dans ce manuscrit, le terme connaissance se rapporte au contexte de la localisation de robots mobiles, sauf mention contraire.

III.2 Sources de connaissance

La connaissance résulte du traitement de l'information. Cette information peut provenir de différentes sources, c'est l'objet de cette partie. Nous intégrons dans une même partie données et informations. Dans ce cadre, un robot s'apparente très fortement à un humain et il est possible d'avoir une forte analogie entre les sources de la connaissance robotique et les sources de la connaissance humaine comme le montre le tableau III.1.

Sources de connaissances Robot		***Sources de connaissance Humain***
Contraintes applicatives (exemple : rouler sur la route) …	**Information innée**	Parents École …
Cartes	**Information innée/acquise**	Mémoire Livres …
Autres robots Cloud Humain …	**Information transmise par des tiers (acquise)**	Pairs Internet …
Odométrie Angle au volant Centrale inertielle …	**Capteurs proprioceptifs (information acquise)**	Oreille interne …
Caméra Télémètre laser Radar …	**Capteurs extéroceptifs (information acquise)**	Vue Toucher Odorat …

Statique

Dynamique

TABLE III.1 – Analogie entre les sources de la connaissance humaine et robotique classées selon leur dynamique

Ces diverses sources de connaissances peuvent être classées suivant leur aspect dynamique. En effet, si certaines informations sont totalement statiques et n'évoluent donc pas dans le temps (cas de la connaissance innée), d'autres au contraire seront dynamiques et peuvent être amenées à changer au fur et à mesure. Certaines informations peuvent être à la frontière entre le statique et le dynamique comme les cartes par exemple.

III.2.1 Information a priori

L'information innée ou a priori est la première information disponible. Elle désigne l'information disponible avant toute utilisation du robot. Cette information peut désigner des choses évidentes comme le fait que le robot se déplace sur la terre et des choses qui ne sont vraies que dans le cadre de notre application comme le fait que le robot se déplace sur la route. Elle peut également concerner l'équipement du robot. Cette information peut être multiple :

- modèle dynamique du robot ;
- environnement d'évolution (climat, type de sol, véhicule sur route ou pas, ...) ;
- types et performances des capteurs ;
- types et performances des détecteurs.

L'utilisation de cette information peut permettre de faciliter l'intégration d'informations plus dynamiques en les replaçant dans leur contexte.

III.2.2 Les cartes

Une source de connaissance est donnée par les cartes qui sont en quelque sorte une photographie de l'environnement à un instant t. Elles sont par nature statiques même si elles peuvent être amenées à évoluer au cours du temps pour être mises à jour. Certaines cartes peuvent également être très dynamique, par exemple pour prendre les obstacles mobiles, comme les grilles d'occupation. Les cartes ont déjà été détaillées dans le chapitre précédent (chapitre II).

III.2.3 Interaction robot/humain, robot/robot, robot/infrastructure

La connaissance peut également venir d'autres sources, comme un humain par exemple. Cela suppose une interaction entre l'humain et le robot [8]. Cette interaction peut être multiple : la parole, des gestes physiques, etc. [59, 133, 140]. Nous ne parlons pas ici de l'interaction qui permet au robot et à l'humain de réaliser de la comanipulation mais de l'interaction qui permet d'échanger des informations. Cela suppose que les informations fournies par l'humain doivent pouvoir être comprises et intégrées par le robot ce qui n'est pas forcément une tâche des plus aisées car un humain peut avoir des expressions, des gestes qui ont un certain sens dans un contexte particulier et signifier quelque chose de fondamentalement différent dans un autre contexte. Le système doit donc être robuste à ce genre d'interprétation et la plateforme d'interaction simple et intuitive pour que l'humain puisse la comprendre et donner des informations pertinentes au robot. Cette information peut être binaire comme l'absence ou la présence d'un panneau de stop sur la route par exemple ou bien être beaucoup plus complexe comme « l'objet est à une distance de 10 mètres et avec un angle de 45 degrés ». Les auteurs qui se sont penchés sur le souci de l'interaction humain/robot font généralement des distinctions assez différentes. Dans [23], l'humain n'est là que pour répondre à une question appelant une réponse binaire ce qui simplifie grandement le processus mais d'un autre côté ne permet pas d'utiliser le plein potentiel d'un humain. En outre, cette façon de faire revient à contraindre l'humain pour être utile au robot alors que le robot est censé être là pour aider l'humain.

Un robot mobile pourra également faire appel à d'autres robots comme source de connaissances. Les robots peuvent s'échanger entre eux des informations sur l'environnement. Dans [86], deux robots sont utilisés, un seul à la fois est en mouvement tandis que l'autre l'observe, calcule son déplacement relatif et le lui fournit, fonctionnant

ainsi comme un capteur proprioceptif. Cette technique peut être utile dans des environnements assez hostiles sans repères visuels et avec beaucoup de glissement comme par exemple sur la neige. Ces échanges de données peuvent engendrer des problèmes de corrélation ou de consanguinité des données [24]. Pour comprendre cela, imaginons le dialogue suivant : soient trois personnes A, B, et C.

```
A dit à B : "La rue est bloquée, il y a peut-être un accident."
B dit à C : "Il y a probablement un accident dans la rue,
  elle est donc bloquée."
C dit à A : "Un accident a vraisemblablement eu lieu dans la rue"
A dit à B : "J'ai eu une confirmation,
 il y a bien eu un accident dans la rue."
```

L'information qu'a reçu A la seconde fois n'était pas nouvelle puisque c'était lui qui l'avait donnée précédemment. Il a donc renforcé son hypothèse initiale sans apport d'information supplémentaire puisque ces données sont corrélées. Le même phénomène peut se produire lors d'une communication entre plusieurs robots s'il n'est pas pris en compte.

III.2.4 Les capteurs

Une grande source de connaissance est issue de ce que nous percevons de nous-mêmes ou des choses extérieures. Cette information est donnée par les capteurs. Cette source de connaissance est forcément très dynamique, les données sont actualisées régulièrement à la vitesse fixée par la fréquence de fonctionnement du capteur. La grande famille des capteurs peut être divisée en deux sous-groupes en robotique mobile : les capteurs proprioceptifs et les capteurs extéroceptifs. La proprioception, comme son nom l'indique, consiste en la perception propre, la perception de soi. Les capteurs proprioceptifs sont donc les capteurs qui mesurent directement une valeur sur l'état interne du robot. À titre de comparaison, en fermant les yeux, un humain arrive malgré tout à savoir si son bras est en train de bouger ou non. Même s'il ne bouge pas par lui-même (par exemple dans une voiture en mouvement), l'oreille interne de l'humain lui permet de mesurer de manière approchée son déplacement relatif. Dans le cadre de la robotique mobile, les capteurs proprioceptifs mesurent généralement le mouvement du robot. A contrario, les capteurs extéroceptifs mesurent des données sur l'environnement extérieur (perception extérieure). Nous présentons ici les principaux capteurs utilisés en robotique mobile.

Les capteurs proprioceptifs

Capteurs inhérents au véhicule L'odométrie qui signifie étymologiquement « mesure du voyage » permet de mesurer le mouvement des roues. Ce type de capteur possède l'avantage d'être bon marché et d'être présent sur presque tous les véhicules produits aujourd'hui car il est notamment utile pour les systèmes d'aide à la conduite comme l'ABS (Anti-Blocage de Sécurité des roues en cas de freinage brusque), ou l'ESP (correcteur électronique de trajectoire). Son utilisation à grande échelle en a fait un capteur bon marché et incontournable. Avec uniquement le mouvement des roues, il est possible de connaître le déplacement global de la voiture y compris dans les tournants. Cependant les mesures risquent de diverger rapidement notamment à cause du glissement des roues sur le sol, aussi les mesures odométriques sont généralement couplées avec une mesure d'angle de braquage des roues.

Capteurs inertiels L'inertie d'un corps désigne sa résistance à un changement de vitesse. Les capteurs inertiels sont basés sur ce principe et mesurent les accélérations (accéléromètres) ou les rotations (gyromètres). Une centrale inertielle est classiquement constituée de 3 gyromètres et de 3 accéléromètres. Il est ainsi possible d'avoir les mêmes informations qu'avec des capteurs odométriques et/ou d'angle de braquage mais de manière augmentée. En effet, l'information est en 3 dimensions avec les différents angles possibles, à savoir le roulis, le tangage et le lacet (figure II.1). Comme pour les capteurs inhérents au véhicule, les capteurs inertiels sont sujets à l'accumulation d'erreurs dans le temps. Cependant, le marché initial de ce type de capteurs étant l'aviation ou l'armée (spécialement les missiles), des centrales inertielles possédant une dérive très faible existent mais à un coût prohibitif pour une application aux véhicules terrestres accessibles au grand public. Les performances seront ainsi très variables suivant les modèles.

Les capteurs extéroceptifs

La caméra La caméra est devenue un capteur très populaire et utilisé dans des applications très diverses (visio-conférence, surveillance de lieux publics, contrôle non destructif, jeux vidéos, ...). La caméra peut souvent être considérée comme l'œil du robot dans le sens qu'elle apporte la même information que celle de l'œil chez l'humain. Les caméras ont un faible coût, apporte beaucoup d'informations sur l'environnement et sont de plus en plus présentes sur les voitures aujourd'hui. En contrepartie, elles sont toutefois sujettes à un certain nombre de contraintes. Si des mesures géométriques doivent être réalisées, elles doivent être calibrées [89]. Elles sont en outre fortement assujetties aux conditions climatiques, une pluie intense, un soleil aveuglant, la présence de neige risquent de perturber de façon non négligeable voire rendre inopérante la caméra. Elles peuvent également être sujettes à des défauts comme le flou, la distorsion, le temps de prise de vue, la saturation, etc. Il existe des caméras plus perfectionnées comme les caméras RGBD [56] qui fournissent également une carte de profondeur.

Le télémètre laser Le télémètre laser fournit directement la distance d'un objet par rapport au capteur. Un faisceau laser est émis et réfléchi par les objets environnants, le temps de vol est alors mesuré et permet de calculer la distance entre l'obstacle et le capteur. Afin de pouvoir balayer l'environnement, le faisceau laser est émis sur un miroir rotatif permettant de balayer l'espace.

Généralement, les télémètres laser utilisés en robotique fournissent une seule nappe et balayent donc l'espace suivant deux dimensions seulement. La présence d'éléments mécaniques (qui vont donc s'user) et leur coût limitent l'utilisation des télémètres laser dans l'industrie automobile. Des capteurs plus évolués ont vu le jour ces dernières années, notamment le velodyne [131] qui permet d'avoir une vision de 3D à 360° grâce à ses 64 nappes (voir figure III.2) et qui est utilisé dans la Google Car. Ces capteurs ont en revanche un coût assez élevé.

Le radar Développé initialement dans les années 30, le radar a connu un fort développement par la suite. Il possède de nombreux avantages, notamment sa faible dépendance aux conditions météorologiques comparativement à la caméra ou au télémètre laser. Le principe du radar est assez similaire à ce dernier, un émetteur envoie une onde qui est réfléchie par les objets environnants, la mesure de l'écho permet de déterminer la position de l'objet ainsi que sa vitesse. La distorsion d'un radar peut être assez grande

FIGURE III.2 – Vélodyne : A gauche une photo du capteur, à droite un exemple d'image 3D fournie par le vélodyne

durant son déplacement [151]. Des objets fantômes (speckle) peuvent également être détectés par le radar ou au contraire des objets existants peuvent ne pas apparaître dans l'image capteur, ce qui peut rendre l'utilisation du radar assez délicate.

Conclusion sur les capteurs

Nous avons entraperçu ici les différents capteurs utilisés en robotique mobile. Cette liste n'est pas exhaustive, il existe encore d'autres types de capteurs comme le magnétomètre, les télémètres ultra-sons, les caméras à temps de vol, etc. Ce panel de capteurs montre bien la diversité et la richesse des informations qui peuvent être perçues et la nécessité d'avoir un système performant pour les utiliser au mieux.

III.2.5 Bilan des sources de connaissances

Les sources de connaissances peuvent être très variées et fournir des informations différentes. Quand nous parlons d'informations différentes, cela peut être par la nature même des informations, mais également par le contexte environnant, la précision des informations, la valeur de cette information, ... Il est impossible de définir une source qui soit meilleure que les autres dans tous les domaines, chacune possède ses atouts et ses faiblesses qui varient suivant les utilisateurs, et le contexte. Ce qui pourra être indifférent dans un cadre sera une forte faiblesse dans un autre ; par exemple le coût peut être négligeable ou prépondérant suivant l'application visée. Dès lors, comment accéder à ces ressources et surtout comment les exploiter de manière optimale ?

III.3 Exploitation des sources d'information

Dans la partie précédente, nous avons pu voir les différentes sources d'informations possibles pour le robot. Cependant, se pose maintenant le problème d'utilisation des données. La façon d'exploiter les données peut être classée en deux parties : les approches descendantes dites Top-Down et les approches ascendantes Bottom-Up (figure III.3). Les approches Top-Down sont les approches qui partent de l'information connue ou des objectifs recherchés et qui sélectionnent les données utiles fournies par les

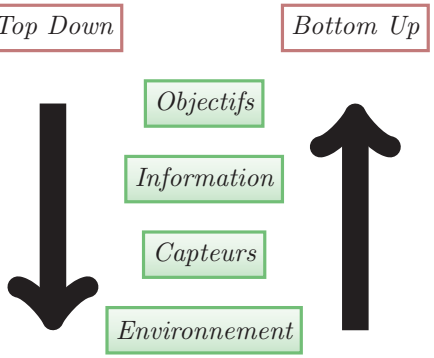


FIGURE III.3 – Illustration des approches Top-Down (à gauche) et Bottom-Up (à droite)

capteurs en fonction de ces objectifs. Les approches Bottom-Up ne se soucient pas initialement de l'objectif ou du contexte mais commencent par récolter toutes les données des capteurs disponibles et vont ensuite les fusionner pour en déduire de l'information qui permettra ou non d'atteindre les objectifs fixés. Certaines études [26, 125] ont démontré que le cerveau utilise les deux modes de fonctionnement.

III.3.1 Les systèmes Bottom-Up

Les approches *Bottom-Up* ou *data-driven* ont été largement utilisées et sont actuellement prépondérantes en robotique. Certains types d'applications doivent comporter une partie Bottom-up car elles concernent la prise en compte d'éléments non connus à l'avance. Nous pouvons citer les méthodes prenant en compte les obstacles [9, 87], la détection des piétons [49, 157], la détection de violation d'intersection [91]. Dans le domaine de la robotique mobile, une étape importante concerne l'utilisation de cartes et donc la construction de carte. Construire une carte de son environnement implique nécessairement une approche Bottom-Up. En effet, pour construire la carte il est nécessaire de détecter des éléments de l'environnement dont on ne sait pas qu'ils existent et sur lesquels aucun a priori n'est disponible. Ces éléments sont alors détectés directement à partir des données capteurs. La construction de cartes est ainsi essentiellement dirigée par les données. C'est pourquoi toutes les techniques de SLAM [39, 42, 93] possèdent une partie Bottom-Up .

Les approches bottom-up possèdent l'avantage de pouvoir détecter des événements qu'il n'était pas possible d'anticiper, en contrepartie elles sont toujours dépendantes des capteurs et sont incapables de prédire l'avenir. Elles peuvent également engendrer un coût de calcul important spécialement dans les systèmes multi-capteurs par le fait que toutes les données capteurs sont utilisées et analysées.

III.3.2 Les systèmes Top-Down

Généralités sur les méthodes Top-Down

Un processus Top-Down peut être décrit comme un système dirigé par l'information (en anglais information-driven) par opposition à un processus Bottom-Up qui sera dirigé par les données (en anglais data-driven). Les processus Top-Down permettent une sélection des données pertinentes et facilitent la distinction bruit/signal. Par exem-

ple, c'est grâce à un processus Top-Down que le cerveau humain arrive à suivre une conversation avec une autre personne dans un environnement bruyant et où différentes conversations ont lieu. Il permet au cerveau humain de focaliser toute son attention sur ce qui l'intéresse et de ne pas se laisser perturber ou sinon dans une moindre mesure par ce qu'il ne juge pas digne d'intérêt. Un système Top-Down ne se contente pas de voir, il regarde, c'est-à-dire qu'il est actif. Le cerveau sait ce qu'il recherche et il va donc mettre en place un système de règles spécifiques mises en place par le cortex frontal [125]. Également un processus Top-Down implique de prévoir ce qui va se passer ou plus exactement ce qui est dans l'ordre des probables à partir de l'information précédente. Cela a deux grands impacts en permettant d'une part de sélectionner les données de manière efficace comme nous venons de le voir et d'autre part de pouvoir anticiper la réponse et de planifier le futur [116]. Cette anticipation du futur permet de ne plus rester passif face aux données et d'avoir une place d'observateur comme dans un système Bottom-Up mais de devenir un acteur à part entière du processus [117]. Ceci est valable pour les humains mais également pour un système robotique.

Application concrète : le sensor management

Le sensor management consiste à utiliser les capteurs les plus pertinents dans une situation donnée et suivant les objectifs fixés. Ce genre de méthode, en plein développement, existe spécialement dans le tracking de cibles [57]. Nous pouvons définir d'une manière large le sensor management comme la mise en place d'une politique de gestion de capteurs permettant de calculer à chaque instant la configuration optimale des capteurs sous certaines contraintes définies par l'application et le contexte.

Il est possible de distinguer deux grandes familles de techniques de sensor management : les techniques dirigées directement par la tâche à accomplir et les techniques pilotées par l'information. Les premières sont faciles à comprendre, le principal critère pour sélectionner un capteur est directement le but de l'application. Les secondes ne prennent pas en compte directement le but mais plutôt l'information globale, le but étant de maximiser l'information [11]. Les techniques basées sur l'information sont nombreuses. La première tâche à réaliser est d'avoir un moyen de quantifier l'information globale. Pour cela plusieurs formalismes ont été proposés dans la littérature : optimisation de l'information de Fisher [73], divergence de Kulback-Leibler [96], entropie de Rényi [83], etc. Les domaines où ce genre de technique a été appliqué sont nombreux : fusion d'information multi-capteurs [154], tracking de cibles avec des réponses de capteurs incertaines [79], tracking multi-cibles à l'aide d'un réseau dynamique de capteurs [84], planification du chemin d'un robot [155], contrôle actif d'une caméra pour la reconnaissance d'objets [40], etc.

III.3.3 Synthèse des approches Top-Down et Bottom-Up

Comme nous venons de le voir, chacune des deux approches possèdent des avantages et des inconvénients. Une approche purement Top-Down ne pourra jamais par exemple détecter qu'un piéton passe devant le robot, dans le même temps une approche purement Bottom-Up ne pourra pas anticiper. Dans le cas d'un animal ou d'un humain, il a été prouvé que le cerveau utilisait une combinaison de processus Top-Down et Bottom-up. [26] montre le résultat d'expériences visant à observer les activités neuronales correspondant à l'un ou à l'autre des processus afin notamment d'en déduire quelles sont les parties du cerveau activées, les fréquences utilisées, etc. Il semblerait que les meilleures performances soient réalisées à l'aide de systèmes combinant les

deux types de méthodes. Bien évidemment, ce résultat est global et il est toujours possible de rencontrer des situations particulières où ce constat devient faux. Il est également à noter que ce genre de système ne dit absolument rien sur la façon dont l'information sera traitée par la suite mais seulement sur la manière de l'obtenir.

III.4 Gestion des informations pour en extraire de la connaissance

Nous avons vu dans les parties précédentes quelles étaient les sources de la connaissance et comment y accéder. Une fois ces étapes réalisées, il faut maintenant utiliser au mieux les données récoltées et en soutirer le maximum d'information pour l'application visée. Que la connaissance ait été acquise de manière Bottom-Up ou Top-Down, le problème demeure, comment agréger ces différentes informations ? Diverses méthodes ont été développées pour gérer au mieux les informations provenant des capteurs et prendre une décision efficace. Nous en présentons ici principalement trois qui sont ou ont été parmi les plus populaires et qui ont fait leurs preuves, à savoir les systèmes experts, les réseaux bayésiens et les réseaux de neurones.

III.4.1 Définitions

Dans cette section, nous définissons certains termes très usités.

Évidence Le terme anglais « evidence » signifie que l'on connaît la réalisation d'un événement (par exemple « le jardin est mouillé »). Avoir une évidence signifie que nous sommes capables de savoir si tel événement est vrai ou faux pour un événement binaire ou de connaître sa valeur pour un autre type d'événement ;

Inférence Suite à une évidence, nous pouvons avoir de la connaissance sur d'autres événements. Par exemple, si notre jardin est mouillé alors la probabilité qu'il ait plu durant la nuit augmente, s'il est sec elle sera nulle. Ce processus de déduction s'appelle l'inférence.

III.4.2 Systèmes experts

« *Un système expert est un programme conçu pour simuler le comportement d'un humain qui est un spécialiste ou un expert dans un domaine très restreint* » P.Denning (1986).

Les systèmes experts ont été créés dans le but de reproduire le comportement d'un expert humain dans une situation très précise. Contrairement aux réseaux de neurones, ils ne cherchent pas à reproduire le comportement du cerveau humain mais utilisent seulement des heuristiques tirées de l'expérience des experts du domaine. Nous pourrions dire qu'un système expert agit de manière diamétralement opposée à celle d'un réseau de neurones. En effet un réseau de neurones peut être vu comme une boîte noire dont il est délicat de prédire le comportement tandis que pour un système expert chaque règle est explicitée et compréhensible par un humain.

Définition III.1 (Systèmes experts)
On appelle système expert l'ensemble de deux parties :
- *une base de connaissance constituée de faits et de règles, cette base contient la connaissance des experts du domaine ;*

– *un moteur d'inférence, ce moteur utilise la connaissance du domaine et l'information fournie afin de fournir une solution experte.*

Les systèmes experts pourront donc s'appliquer uniquement suivant des règles bien précises. Les systèmes experts utilisent d'une part les données tirées des expérimentations, des règles et un moteur d'inférence. Ces systèmes sont notamment connus depuis le développement dans les années soixante-dix de MYCIN dans le milieu médical [25]. Les systèmes experts sont originellement conçus pour fonctionner dans des environnements déterministes certains. Afin de les utiliser dans un contexte incertain, des méthodes ont été déployées notamment par utilisation de logique floue [72]. Malgré tout, cela pose certains inconvénients et oblige à utiliser des hypothèses contraignantes « d'indépendance conditionnelle et de modularité de la connaissance » [38]. Après avoir connu une forte utilisation, les systèmes experts sont aujourd'hui en perte de vitesse et sont de moins en moins utilisés pour deux principales raisons : leur faible capacité d'apprentissage même si des travaux ont été effectués en ce sens [47] et leur mauvaise gestion de l'incertitude.

III.4.3 Réseaux bayésiens

Les réseaux bayésiens sont une méthode basée sur la théorie des graphes et sur la théorie des probabilités. Ils permettent une représentation graphique de la connaissance simple et intuitive : les causes (modélisés par des nœuds) sont reliées aux effets par une flèche. Par exemple sur la figure III.4, le fait d'être myope influe sur le fait de porter des lunettes ou non. La myopie étant une des causes amenant à porter des lunettes, une flèche partant de l'événement « Myopie » arrive sur l'événement « Lunettes ».

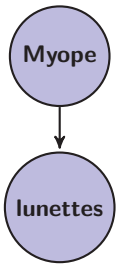


FIGURE III.4 – Exemple simple de réseau Bayésien

A ce graphe, nous venons associer la théorie des probabilités. Un réseau bayésien sera alors capable de calculer différentes probabilités : quel est le pourcentage de la population à être à la fois myope et à porter des lunettes ? Quel est le pourcentage de la population à porter des lunettes ?

Théorie des probabilités

Afin de comprendre le concept de réseau bayésien, il est nécessaire de présenter quelques bases de la théorie des probabilités. La théorie des probabilités est un système mathématique permettant de formaliser le degré de certitude que l'on a dans un événement.

Définition III.2 (Espace probabilisé ou espace de probabilité)
Un espace probabilisé ou espace de probabilité est un triplet (Ω, Z, P) *où :*
- Ω *est l'espace des résultats possibles d'une épreuve aléatoire ;*
- Z *est une tribu de parties de* Ω*. Ses éléments sont appelés des événements ;*
- P *est une mesure de probabilité, c'est-à-dire une fonction* $P : Z \rightarrow [0,1]$ *ayant les deux propriétés suivantes (appelées axiomes de Kolmogorov) :*
  - ✧ $P(\Omega) = 1$
  - ✧ $P(\bigcup_{i\in I} Z_i) = \sum_{i\in I} P(Z_i)$ *avec* $\forall i, j, (i \neq j) \Longrightarrow (Z_i \bigcap Z_j = \varnothing)$

Quelques définitions sont données ici :
- $P(X = x|Y = y)$ signifie probabilité que l'événement X ait la valeur x sachant que l'événement Y a la valeur y, c'est la probabilité conditionnelle ;
- $P(X = x, Y = y)$ aussi noté $P((X = x) \wedge (Y = y))$ signifie probabilité que l'événement X ait la valeur x et que l'événement Y ait la valeur y, c'est la probabilité jointe.

Rigoureusement chaque probabilité devrait être notée sous la forme $P(X = x)$ où X est l'événement et x est la réalisation de cet événement. Cependant cette notation est simplifiée et seulement notée sous la forme $P(X)$. Par exemple, si nous considérons deux événements binaires A et B pouvant être vrais ou faux et tels que $P(A|B) = P(A)$, cela signifie que :

$$\begin{array}{rcl} P(A = vrai|B = vrai) & = & P(A = vrai) \\ P(A = vrai|B = faux) & = & P(A = vrai) \\ P(A = faux|B = vrai) & = & P(A = faux) \\ P(A = faux|B = faux) & = & P(A = faux) \end{array}$$

Définition III.3 (Probabilité Conditionnelle)
Soient X *et* Y *deux événements, il est possible de définir la probabilité conditionnelle de l'événement* X *sachant* Y *notée* $P(X|Y)$ *:* $P(X|Y) = \dfrac{P(X,Y)}{P(Y)}$

A partir de cette équation, nous pouvons définir un certain nombre de termes :
- deux événements X, Y sont dits *indépendants* si et seulement si $P(X|Y) = P(X)$;
- soient trois événements X, Y et Z, X et Y sont dits *indépendants conditionnellement* à Z si et seulement si $P(X|Y,Z) = P(X|Z)$ et $P(Y|X,Z) = P(Y|Z)$. En d'autres termes, si Z est connu, connaître Y ne nous apportera pas plus d'informations sur X et inversement.

Théorème 1 (Probabilité totale)
Considérons un système exhaustif d'événements, c'est-à-dire un ensemble au plus dénombrable $A_1, A_2, \ldots$ *d'événements deux à deux incompatibles, tels que* $\bigcup_{i\in I} A_i = \Omega$*. Alors :* $\forall B \in \Omega, P(B) = \sum_{i\in I} P(A_i \bigcap B) = \sum_{i\in I} P(A_i)P(B|A_i)$

Théorème 2 (Théorème de Bayes)
La probabilité conjointe de deux événements X,Y peut être calculée suivant la règle de conjonction :

$$\begin{aligned} P(X,Y) &= P(X)P(Y|X) \\ &= P(Y)P(X|Y) \end{aligned} \tag{III.1}$$

Cette forme peut s'écrire également de la façon suivante :

$$P(X|Y) = \frac{P(X)P(Y|X)}{P(Y)} \tag{III.2}$$

Le théorème de Bayes peut s'écrire de manière plus générale de la manière suivante :

Soit$\{x_1,x_2,\ldots,x_k,\ldots,x_n\}$ un ensemble d'événements, alors

$$\forall i \in |[1,n]|, P(x_i|x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = \frac{P(x_i)P(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n|x_i)}{P(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)} \tag{III.3}$$

Définition III.4 (Probabilité marginale)
Marginaliser la probabilité d'un événement Y revient à calculer $P(Y)$: $\sum_X P(X,Y) = P(Y)$ La probabilité d'un événement seul sera appelée la probabilité marginale de cet événement ou encore probabilité a priori car elle est définie sans prendre en compte les observations ultérieures.

Théorie des graphes

Le cadre général de la théorie des probabilités ayant été posé, l'autre aspect des réseaux bayésiens, à savoir la théorie des graphes, est maintenant abordé.

Définition III.5 (Graphe simple)
Un graphe simple G est défini par un couple (V,E) où V est l'ensemble des nœuds de G, et E est un ensemble de paires d'éléments de V appelées les arcs de G.

Par exemple, la figure III.5 représente un graphe simple $G = (V,E)$ avec $V = \{A,B,C,D\}$ et $E = \{(A,B),(A,C),(B,C),(B,D),(C,D)\}$.

Définition III.6 (Graphe simple orienté)
Un graphe simple $G = (V,E)$ est dit orienté quand les éléments de E ne sont pas des paires mais des couples, c'est-à-dire que l'ordre est important. Le premier élément d'un couple est appelé parent ou cause du second.

La figure III.6a représente un graphe simple orienté $G = (V,E)$ avec $V = \{A,B,C,D\}$ et $E = \{(A \to B),(A \to C),(B \to D),(C \to B),(D \to C)\}$.

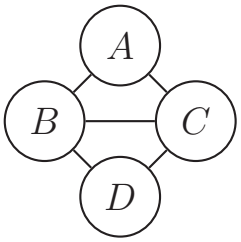
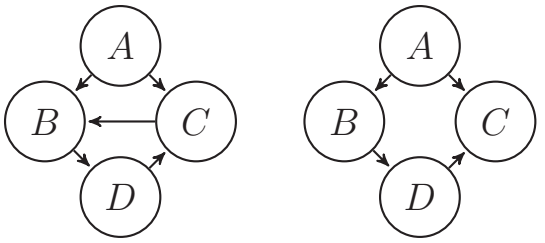


FIGURE III.5 – Représentation d'un graphe simple G



(a) Graphe simple orienté avec circuit

(b) Graphe simple orienté sans circuit

FIGURE III.6 – Graphes simples orientés

Définition III.7 (Chemin et circuit)
Dans un graphe orienté, un chemin d'origine A et d'extrémité B est défini par une suite finie d'arcs consécutifs, reliant A à B, c'est-à-dire tel que l'origine du premier arc est A, l'extrémité du dernier est B et chaque extrémité des autres arcs correspond à l'origine de l'arc suivant. On appelle circuit un chemin dont l'extrémité et l'origine sont confondus.

Utilisation par un réseau bayésien

La théorie des graphes et la théorie des probabilités ayant été présentées, il est maintenant possible de définir mathématiquement les réseaux bayésiens.

Définition III.8 (Réseau bayésien)
Un réseau bayésien est défini par :
- *un graphe simple orienté sans circuit* $G = (V, E)$ *;*
- *un espace probabilisé fini* (Ω, Z, P) *;*
- *un ensemble de variables aléatoires associées aux nœuds du graphe et définies sur* (Ω, Z, P) *tel que :*

$$P(V_1, V_2, \ldots, V_n) = \prod_{i=1}^{n} P(V_i | C(V_i))$$

où $C(V_i)$ est l'ensemble des causes de V_i dans le graphe G.

D'un point de vue applicatif, nous pouvons considérer un réseau bayésien comme étant un arbre de calcul de probabilités. Le calcul des probabilités dans un réseau bayésien se fait en s'appuyant sur la théorie des graphes, plusieurs algorithmes ont été développés dans la littérature à ce sujet. L'inférence est également prise en compte dans ce type de réseau permettant de mettre à jour le réseau au fur et à mesure que des observations sont disponibles. De par son formalisme bayésien, un réseau bayésien est naturellement adapté pour l'étude d'événements incertains de nature.

Nous faisons ici le choix de ne parler que des réseaux bayésiens à variables discrètes, c'est-à-dire où les événements ne peuvent prendre qu'un nombre fini de valeurs même s'il existe des réseaux bayésiens avec des valeurs continues (il est possible de formaliser le filtre de Kalman comme un réseau bayésien) [41] ou même avec un mélange de valeurs continues et discrètes dans le même réseau [105].

III.4.4 Réseaux de neurones

Puisqu'ici le but est d'utiliser et gérer au mieux de l'information pour en tirer de la connaissance, le meilleur exemple que nous puissions trouver est le cerveau humain qui est capable d'analyser de l'information très hétérogène, de s'adapter en prenant en compte de nouveaux événements, d'apprendre de ses erreurs, etc. L'élément de base du cerveau étant le neurone, les chercheurs se sont penchés sur son fonctionnement et ont essayé en partie de le reproduire. Ce principe a donné naissance aux méthodes neuromimétiques. Un réseau de neurones est constitué de plusieurs entités indépendantes reliées entre elles par des actions de stimulation ou au contraire d'inhibition.

Définition III.9 (Neurone formel)
On appelle neurone formel la représentation mathématique d'un neurone biologique. Un neurone formel est une fonction paramétrée, de plusieurs variables appelées entrées du neurone ou encore stimuli externes ; la valeur de la fonction est disponible en sortie du neurone, elle est aussi appelée potentiel du neurone. Ce neurone est souvent représenté comme un nœud dans un graphe orienté sur lequel arrivent différents arcs et d'où en part un seul arc (voir figure III.7).

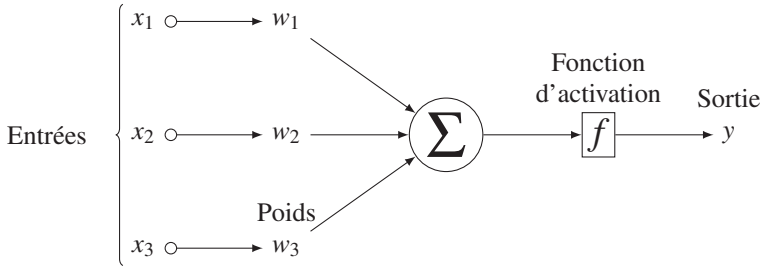


FIGURE III.7 – Représentation du fonctionnement d'un neurone

Définition III.10 (Réseaux de neurones)
Un réseau de neurones est un graphe orienté où les nœuds sont les neurones formels et les arcs donnent les entrées et les sorties des neurones. Les neurones sont généralement organisés sous forme de couches, au sein de chaque couche chaque neurone est potentiellement indépendant des autres et réagit différemment aux stimuli externes. Cet ensemble de couches est appelé réseau de neurones. Ces couches peuvent comporter des nœuds visibles (entrées ou sorties du système) ou invisibles depuis l'extérieur.

La figure III.8 montre un réseau composé de trois couches, deux visibles et une cachée.

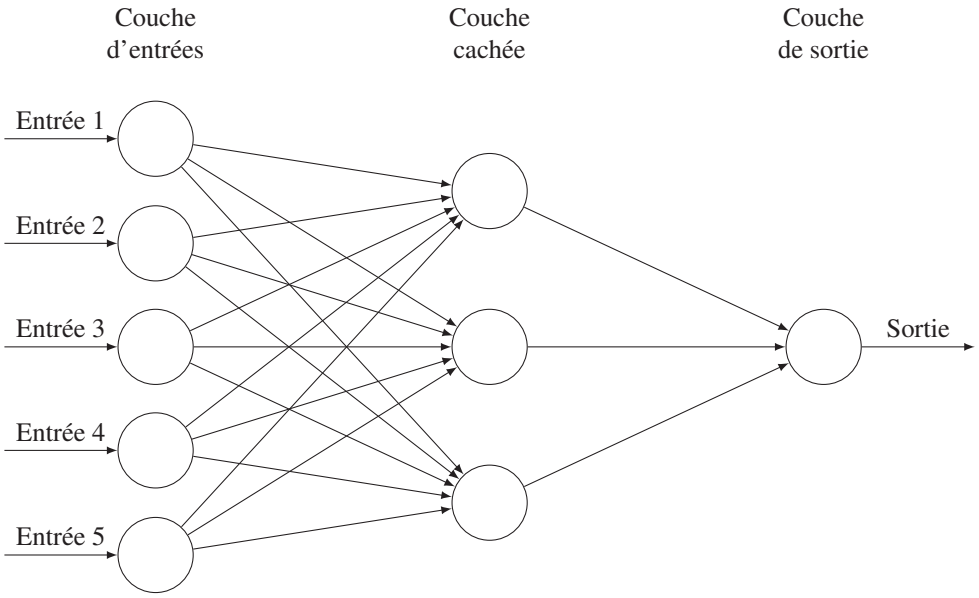


FIGURE III.8 – Exemple d'un réseau de neurones composé de plusieurs couches

Un grand nombre d'applications ont été abordées à l'aide de réseaux de neurones avec un certain succès comme la reconnaissance d'écriture [32], des problèmes de classification, problèmes médicaux [37, 124], localisation de véhicules [121], navigation de véhicules [111, 156]. Au fil des années un grand engouement pour cette méthode de modélisation a été observé et de nombreux modèles ont été créés. L'intérêt principal des réseaux de neurones réside généralement dans leur capacité d'apprentissage. En effet, le réseau apprend les différentes pondérations w_i à appliquer aux neurones à partir d'une base d'apprentissage. Au final un système modélisé par un réseau de neurones se présente comme une boîte noire qui fonctionne sur des cas appris mais qui n'aide pas à comprendre le phénomène qui se passe. Cela peut être très contraignant car si un utilisateur désire rajouter l'influence de tel ou tel événement non pris en compte

précédemment, il est obligé de repasser par une phase d'apprentissage et ne peut pas utiliser le précédent. Il est également difficile d'injecter dans un tel type de réseau de la connaissance a priori même si des travaux ont été réalisés en ce sens [31].

III.4.5 Autres systèmes de représentation de la connaissance

Nous avons présenté les systèmes de représentation de la connaissance les plus connus et/ou les plus génériques. Nous présentons ici deux autres façons de représenter l'information connue : L'analyse de données et les arbres de décision.

Analyse de données

L'analyse de données est un processus de réduction, de transformation et de modélisation de données avec l'objectif de découvrir de l'information utile, d'en tirer des conclusions et finalement de pouvoir effectuer de la prise de décision. L'analyse de données est basée sur les statistiques et intervient sur des grands jeux de données fournis. Elle est également utilisée pour réaliser de la prédiction ou de la classification d'événements. Nous pouvons citer l'Analyse en Composantes Principales (ACP) [66] qui permet de déterminer la redondance entre les données et de travailler avec de nouvelles données indépendantes. Tandis que les réseaux bayésiens peuvent permettre d'identifier les causes d'un événements, l'Analyse en Composantes Principales permet seulement d'observer des corrélations entre les données sans lien de causalité.

Arbres de décision

Un arbre de décision [120] est un outil de décision qui utilise un graphe de représentation des données avec à chaque feuille la décision à appliquer. Un arbre de décision permet de classer un objet à l'aide de questions : chaque nœud de l'arbre représente une question, chaque lien est une réponse à la question et chaque feuille est une classe. Un exemple d'arbre de classification est donné sur la figure III.9. Les arbres de décision possèdent l'avantage d'être rapides et lisibles.

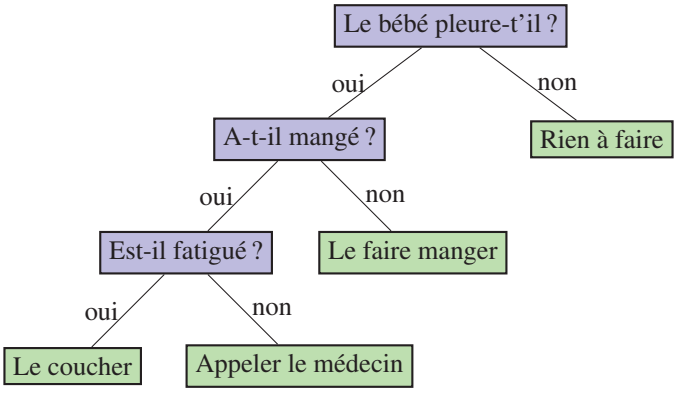


FIGURE III.9 – Exemple d'un arbre de décision

Connaissances	*Analyse de données*	*Réseaux neuronaux*	*Arbres de décision*	*Systèmes experts*	*Réseaux bayésiens*
ACQUISITION					
Expertise seulement				★	
Données seulement	+	★	+		+
Mixte	+	+	+		★
Incrémental		+			★
Généralisation	+	★	+		+
Données incomplètes		+			★
REPRÉSENTATION					
Incertitude				+	★
Lisibilité	+		+	+	★
Facilité		+	★		
Homogénéité					★
UTILISATION					
Requêtes élaborées	+			+	★
Utilité économique	+	+			★
Performances	+	★			

TABLE III.2 – Avantages comparatifs des réseaux bayésiens (tiré de [106])

III.4.6 Récapitulatif

Les méthodes décrites ci-dessus ont été vues à chaque fois de manière individuelle mais cela ne signifie pas qu'elles ne peuvent avoir de connexion entre elles. Par exemple des relations formelles ont pu être montrées entre les réseaux bayésiens et les arbres de décision ou les réseaux de neurones [106]. Dans [67], une équivalence formelle entre les réseaux bayésiens et les réseaux de neurones a été proposée. Toutefois, ces connexions ne sont pas encore opérationnelles et restent pour l'instant assez formelles. Il est possible de comparer les avantages et les inconvénients de chacune des méthodes présentées ci-dessus.

Le tableau III.2 donne une vue d'ensemble des avantages et inconvénients de chacune des méthodes. La représentation adoptée est la suivante :

- à chaque ligne correspond une caractéristique, qui peut être un avantage , ou la prise en compte d'un problème spécifique ;
- si la technique considérée permet de prendre en compte ce problème ou présente cet avantage, un signe + est placé dans la case correspondante ;
- un signe ★ est placé dans la case de la meilleure technique du point de vue de la caractéristique considérée.

Ce tableau permet notamment de déterminer la meilleure des techniques à adopter suivant l'application visée. Du point de vue de la performance, la meilleure technique est l'utilisation de réseaux de neurones, en revanche la gestion de l'incertitude est mieux gérée par les réseaux Bayésiens, etc.

III.5 Présentation de notre approche

Au terme de cette analyse, il nous faut maintenant définir notre approche d'exploitation de l'information en vue de générer la connaissance de la localisation de notre robot. Pour cela, il est nécessaire de définir quel est notre objectif, quels sont les moyens disponibles et par conséquent quelle est la meilleure façon de les exploiter.

Nous désirons développer une application de localisation exploitant des cartes déjà existantes et disponibles facilement. L'application développée doit être générique et pouvoir s'adapter au robot utilisé et notamment aux différents capteurs présents sur le véhicule. Elle doit donc être nativement multi-sensorielle. L'application est destinée à être utilisée sur des véhicules en temps-réel, la méthode ne doit donc pas engendrer un coût calculatoire trop important. Enfin cet algorithme doit pouvoir être utilisé pour réaliser du guidage automatique de véhicules en environnement urbain, ce qui impose une incertitude par exemple inférieure à 10 centimètres et une confiance dans la localisation de l'ordre de 95%.

III.5.1 Choix de la méthode pour accéder à l'information

L'application développée doit savoir s'adapter aux conditions environnementales, aux différents types de capteurs, de détecteurs et d'amers disponibles. Par exemple, en environnement intérieur, nous ne voulons pas que notre système tente d'utiliser le GPS ou de détecter des arbres dans une image capteur qui en est forcément dépourvue. De ce niveau-là, une approche Top-Down permettant de sélectionner la meilleure action à effectuer selon le contexte semble s'imposer.

Une approche Bottom-Up serait utile si nous cherchions à détecter des objets de notre environnement dont nous ne connaîtrions pas l'existence ou la position (comme des piétons ou des amers non-répertoriés). Nous ne sommes pas dans cette situation puisque nous partons ici d'une carte déjà existante et que nous nous limitons à la localisation dans cette carte sans chercher à la construire ou à l'enrichir. En outre, avec une méthode Bottom-Up le problème d'association de données se poserait de manière plus importante qu'avec une approche Top-Down car une méthode Top-Down permet la focalisation sur l'objet recherché lors de la phase de détection. C'est pourquoi, nous privilégions ici l'utilisation d'une approche Top-Down, c'est-à-dire la sélection de l'action à réaliser à chaque instant en fonction des données disponibles, de nos objectifs (détaillés dans le chapitre IV) , et de notre état actuel.

III.5.2 Choix de la méthode pour agréger l'information

Les données que nous voulons traiter ne sont pas parfaites. Nous avons affaire à des capteurs fournissant des données entachées de bruit, des détecteurs imparfaits, des incertitudes sur notre propre position, etc. Ceci nous oblige à utiliser un système permettant de gérer au mieux l'incertitude ce qui est nativement réalisé par un réseau bayésien contrairement aux autres méthodes présentées. De plus, nous désirons avoir un système capable d'évoluer facilement, c'est-à-dire qui nous laisse la possibilité de rajouter la prise en compte d'événements au cours du temps de manière aisée, intuitive et sans nous obliger à repenser totalement l'architecture globale. Il faut par exemple être capable de rajouter des événements comme la présence d'un autre véhicule dans le champ de vision occultant ainsi partiellement les amers visibles, on pourrait aussi imaginer gérer des dysfonctionnements des capteurs, les problèmes météorologiques, etc. Il faut en outre que le fonctionnement interne soit relativement aisé à comprendre. Cela

a notamment deux impacts : un utilisateur appréhendera mieux un système robotique dont il peut comprendre le fonctionnement, et cela permet également de comprendre plus facilement le comportement du robot lors des diverses manipulations.

Ces différents points nous amènent à choisir un réseau Bayésien pour gérer les différentes données. En effet, comme nous l'avons vu un réseau Bayésien possède les caractéristiques suivantes :

- gestion native des incertitudes ;
- Les évolutions n'ont qu'un impact local limité ;
- Représentation graphique relativement intuitive.

III.5.3 Outil de fusion des données détecteur

Afin de fusionner les différentes données issues des détecteurs, il est nécessaire d'avoir recours à un outil de fusion. Cet outil de fusion doit être capable de prendre en compte les incertitudes des données ainsi que l'hétérogénéité de ces dernières. Pour cela, un filtre de Kalman étendu, décrit dans l'annexe A, est utilisé.

III.5.4 Détection et réparation des erreurs d'association

Bien que l'utilisation d'une méthode Top-Down permette d'optimiser la phase de détection et d'association des données, il peut survenir une fausse association entraînant alors une estimation fausse de la position. Afin de résoudre ce problème, le système développé doit être capable de détecter que la position estimée est fausse, d'identifier la cause de cette erreur et de la réparer dans la mesure du possible. Cette détection est rendue possible par le réseau bayésien qui permet de prendre en compte les différents résultats et de donner les probabilités des différents événements, le système de réparation mis en œuvre s'appuie donc sur ce réseau. Ainsi, il pourra être possible d'inférer la cause de cette erreur, est-ce dû au fait que le robot n'est pas là où il pense, à une erreur du processus de détection, à une occultation, etc.

III.5.5 Bilan

Afin de répondre à la problématique posée, nous proposons dans cette thèse une méthode de localisation où chaque ressource est utilisée en fonction de l'objectif final et de l'état courant suivant un processus Top-Down. Les quatre fondamentaux de notre méthode sont : évaluation de l'objectif, sélection de la meilleure action de perception à réaliser en fonction de cet objectif, perception de l'environnement, mise à jour de l'état suivant le résultat de la perception avec détection des erreurs d'association et réparation le cas échéant. Cela est modélisé par la figure III.10.

Pour cela, des moyens techniques sont mis en place et présentés, à savoir entre autres un réseau Bayésien Dynamique, un filtre de Kalman, et une carte géoréférencée de l'environnement accessible via un Système d'Information Géographique (GIS).

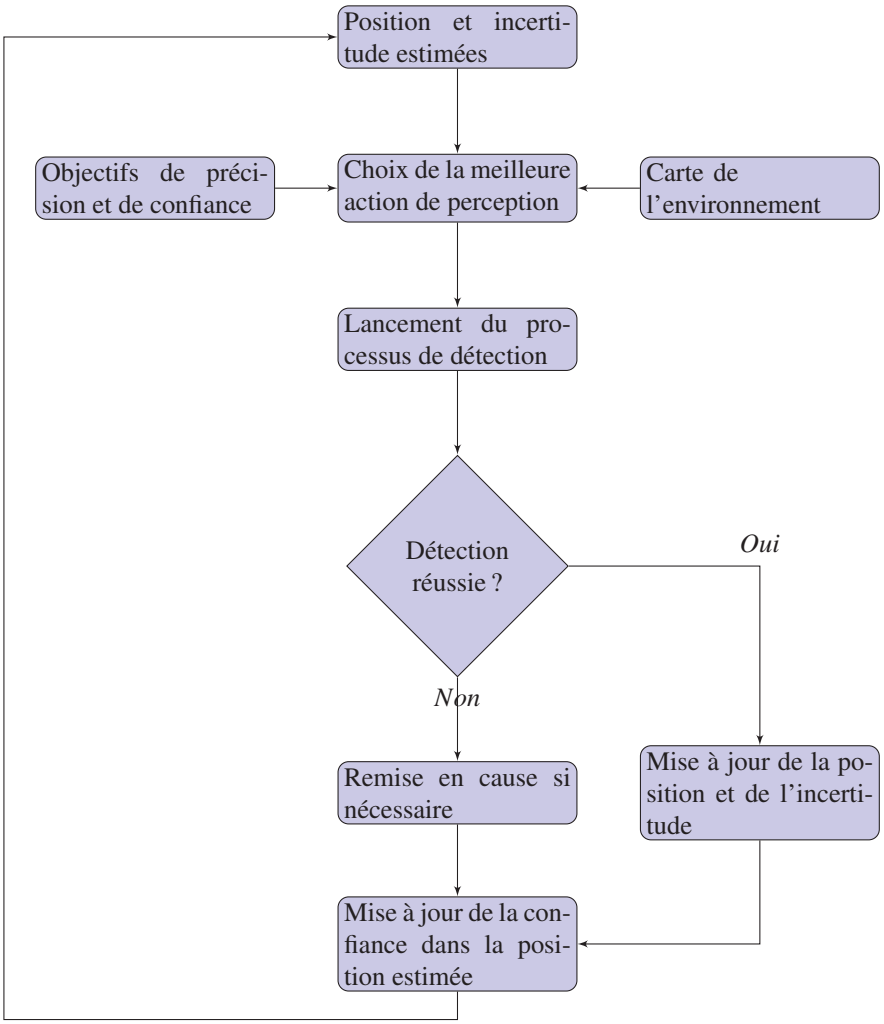


FIGURE III.10 – Algorithme de localisation développé

CHAPITRE IV

Optimisation de la perception

Sommaire

IV.1 Introduction

Afin d'atteindre les objectifs de précision et de confiance définis par notre application, notre système doit sélectionner et utiliser les détecteurs et capteurs disponibles à chaque instant. Pour réaliser ces différentes tâches, nous utilisons la notion de triplet perceptif [143]. Un triplet perceptif T_i est l'ensemble constitué par un amer A_i, un détecteur D_i et un capteur C_i, soit $T_i = \{A_i, D_i, C_i\}$. L'amer est donné par la carte que nous possédons, c'est lui qui nous apporte de la connaissance a priori sur notre environnement (par exemple mur N° 45, arbre N° 23) ; le capteur est dépendant de notre véhicule (positionnement, portée, résolution, ...) et donc de la partie hardware ; le détecteur est lié à la partie software. Le triplet perceptif intègre toutes ces informations et nous fournit tout ce qui est nécessaire pour percevoir notre environnement : il faut comprendre par exemple que pour un même amer (par exemple un poteau), et avec un même capteur (par exemple un lidar) l'algorithme de détection puisse être différent selon la distance de ce poteau au robot. Le choix de ce triplet perceptif doit donc être réalisé le plus intelligemment possible. Des triplets peuvent se révéler sans utilité car leur résultat sera inexploitable ou parce qu'ils n'apportent pas d'information supplémentaire par exemple.

IV.2 Objectifs

Comme cela a été décrit dans le chapitre précédent, le but de nos travaux est de connaître le lieu où se trouve un robot, d'avoir la connaissance de la localisation du robot. Qu'est-ce que la connaissance ? Suivant l'approche classique définie en épistémologie, mise en forme par Platon et régulièrement utilisée depuis, la connaissance est une *croyance vraie et justifiée*. Pour la localisation d'un robot mobile, cette définition est totalement applicable. Il ne s'agit pas seulement de croire que le robot est situé à un endroit précis, il s'agit aussi de pouvoir le justifier. Il est possible que cette justification ne soit pas binaire. En effet nous pourrons peut être seulement le justifier de manière probabiliste, par exemple "Il y a 70 % de chances que le robot soit à tel endroit". L'objectif de notre application va donc être double : avoir une estimation la plus précise possible de la position de notre robot et caractériser la confiance que l'on a dans cette estimation. Cela peut être caractérisé comme un objectif de précision et un objectif d'intégrité ou de fiabilité.

IV.2.1 Objectif de précision

Formalisation

La position de notre robot sera définie par ses coordonnées (x,y) dans le plan cartésien et par son orientation θ. La position du robot étant inconnue pour nous, ces variables sont considérées comme des variables aléatoires gaussiennes. Par conséquent, à chaque instant k le robot sera défini par son vecteur d'état $X_k = (x,y,\theta)^T$ et une matrice de covariance $\mathbf{C}_k$ modélisant l'incertitude sur cette position estimée par le vecteur $\hat{X}_k$. L'objectif de précision pourra être donné par une zone d'incertitude pour le robot, par exemple : 10cm sur x, 2cm sur y et 1° sur θ avec indépendance des valeurs entre elles. Afin de bien comprendre le phénomène, considérons un cas monodimensionnel.

Dans notre cas, l'objectif de précision est un intervalle autour de la valeur réelle dans lequel doit se trouver notre estimation. Cet intervalle appelé « intervalle objectif » sera noté ici I (figure IV.1). A chaque instant, la précision de l'estimation de la

position peut être représentée par une fonction de vraisemblance gaussienne ($p(X_k) = \mathcal{N}(\hat{X}_k, \mathbf{C}_k)$) (voir figure IV.1). La probabilité que la position estimée soit dans cet intervalle est alors égale à la probabilité d'être intègre. Il est alors possible de calculer dans quelle mesure l'objectif a été atteint en calculant le terme P_p correspondant (équation IV.1).

$$P_p = \int_I \mathcal{N}(\hat{X}_k, \mathbf{C}_k) dX \tag{IV.1}$$

On notera ici que dans le cas classique d'utilisation d'un filtre de Kalman, il est supposé que la position est intègre et on a $Pp = \int_{\infty}^{\infty} \mathcal{N}(\hat{X}, \mathbf{C}_k) dX = 1$. Dans notre cas, du fait d'éventuelles erreurs de détection, cette probabilité sera inférieure à 1.

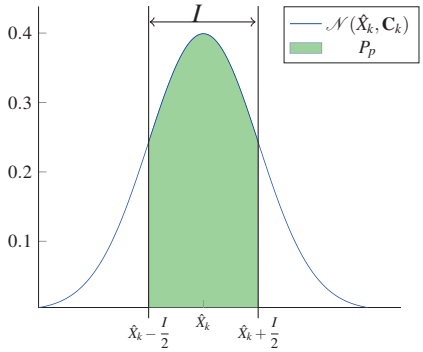


FIGURE IV.1 – Calcul de Pp en dimension 1. I représente notre objectif de précision, la courbe de Gauss représente la fonction de vraisemblance de X_k centrée sur son estimation $\hat{X}_k$

Nous aurons :

$$\begin{aligned} P_p &= \int_I \mathcal{N}(\hat{X}_k, \mathbf{C}_k) dX \\ &= \int_{\hat{X}_k - \frac{I}{2}}^{\hat{X}_k + \frac{I}{2}} \mathcal{N}(\hat{X}_k, \mathbf{C}_k) dX \\ &= \int_{-\infty}^{+\infty} \mathcal{N}(\hat{X}_k, \mathbf{C}_k) \Pi_I(X - \hat{X}_k) dX \end{aligned} \tag{IV.2}$$

où Π_I est la fonction porte définie de la manière suivante :

$$\begin{aligned} \Pi_I : &\mathbb{R} \longrightarrow \mathbb{R} \\ &x \longmapsto \begin{cases} 1 & \text{si x} \in [-\frac{I}{2}, \frac{I}{2}] \\ 0 & \text{sinon.} \end{cases} \end{aligned}$$

L'équation IV.2 est facilement généralisable à un espace à n dimensions. Dans le cas où n est supérieur à 2, le calcul de cette intégrale peut ne pas être simple. Par ailleurs, il peut être intéressant de représenter un intervalle objectif non parallélépipédique. Afin

d'éviter ce souci, la fonction Π, représentant l'intervalle objectif, est remplacée par la fonction suivante qui modélise un intervalle gaussien :

$$\mathbb{R}^n \longrightarrow \mathbb{R}$$

$$X \longmapsto \exp(-\frac{1}{2}(X)\mathbf{C}_I^{-1}(X)^T)$$

Dans cette équation, le terme $\mathbf{C}_I$ correspond à la matrice de variance-covariance désirée. Dans un cas mono-dimensionnel, cette matrice se réduirait par exemple à la variance désirée de notre position estimée. Dans un cas multi-dimensionnel, elle permet de prendre en compte toutes les dimensions et leurs dépendances.

L'équation IV.2 devient alors :

$$\begin{aligned}
P_p &= \int_{-\infty}^{\infty} \exp(-\frac{1}{2}(X-\hat{X}_k)\mathbf{C}_I^{-1}(X-\hat{X}_k)^T).\mathcal{N}(\hat{X}_k,\mathbf{C}_k)dX \\
&= \int_{-\infty}^{\infty} \exp(-\frac{1}{2}X\mathbf{C}_I^{-1}X^T).\mathcal{N}(0,\mathbf{C}_k)dX \\
&= \int_{-\infty}^{\infty} \exp(-\frac{1}{2}X\mathbf{C}_I^{-1}X^T)\frac{1}{\sqrt{(2\pi)^n|\mathbf{C}_k|}}\exp(-\frac{1}{2}X\mathbf{C}_k^{-1}X^T)dX \\
&= \int_{-\infty}^{\infty} \frac{K}{\sqrt{(2\pi)^n|\mathbf{C}|}}\exp(-\frac{1}{2}X\mathbf{C}^{-1}X^T)dX = K
\end{aligned}$$

Avec :

$\frac{K}{\sqrt{(2\pi)^n|\mathbf{C}|}} = \frac{1}{\sqrt{(2\pi)^n|\mathbf{C}_k|}}$ n est la dimension du vecteur d'état (3 dans notre cas)

et $\mathbf{C}^{-1} = \mathbf{C}_k^{-1} + \mathbf{C}_I^{-1}$

Soit :

$$\begin{aligned}
K &= \sqrt{\frac{|\mathbf{C}|}{|\mathbf{C}_k|}} \\
&= \sqrt{\frac{1}{|\mathbf{C}|^{-1}|\mathbf{C}_k|}} \\
&= \sqrt{\frac{1}{|(\mathbf{C}_I^{-1}+\mathbf{C}_k^{-1})\mathbf{C}_k|}} \\
&= \sqrt{\frac{\mathbf{1}}{|\mathbf{C}_I^{-1}\mathbf{C}_k+\mathbf{I}_{n\times n}|}}
\end{aligned}$$

Finalement, la formule de P_p devient :

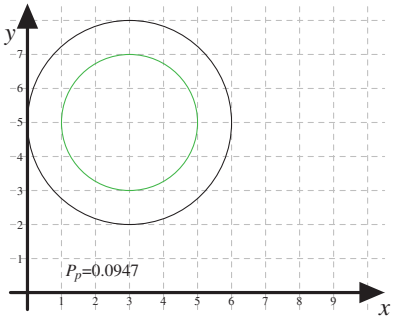
$$\boxed{P_p = \sqrt{\frac{1}{|\mathbf{C}_I^{-1}\mathbf{C}_k+\mathbf{I}_{n\times n}|}}} \tag{IV.3}$$

(IV.4)

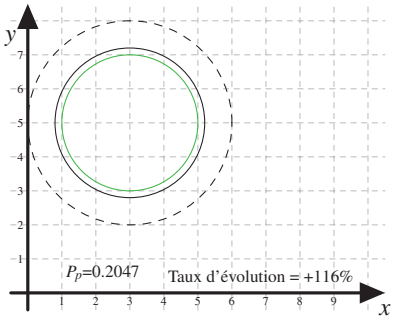
Cet objectif de précision sera un des deux critères pour le choix du triplet à un instant donné. Deux exemples sont donnés ci-après afin d'illustrer le comportement de ce critère de précision.

Exemple 1 : discrimination des triplets en fonction de leur apport vis-à-vis de l'objectif

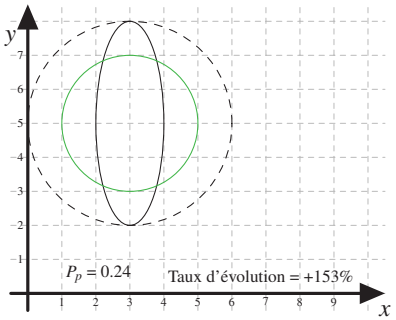
Chaque triplet a un apport potentiel vis-à-vis de l'objectif de précision. À titre d'exemple, prenons la figure IV.2. Deux triplets sont disponibles. Ces deux triplets sont ici seulement connus à travers l'information qu'ils peuvent apporter sur la position et plus particulièrement dans le cas qui nous intéresse sur la précision de la position estimée. L'objectif est de savoir quel triplet utiliser pour optimiser la précision. A priori, il peut sembler délicat de discriminer entre ces deux triplets vis-à-vis de l'objectif de précision. L'un apporte potentiellement de l'information suivant une direction précise (figure IV.2c) tandis que l'autre apporte une information suivant des directions homogènes (figure IV.2b). L'utilisation de notre critère va permettre de réaliser cette discrimination facilement. L'apport P_p est calculé pour chaque triplet comme nous l'avons vu précédemment par la relation IV.3. Dans ce cas de figure précis, c'est le premier triplet qui apporte le plus d'information sur la précision puisque c'est lui qui permet d'arriver à une valeur de P_p plus importante.



(a) État initial



(b) Amélioration potentiellement apportée par le premier triplet

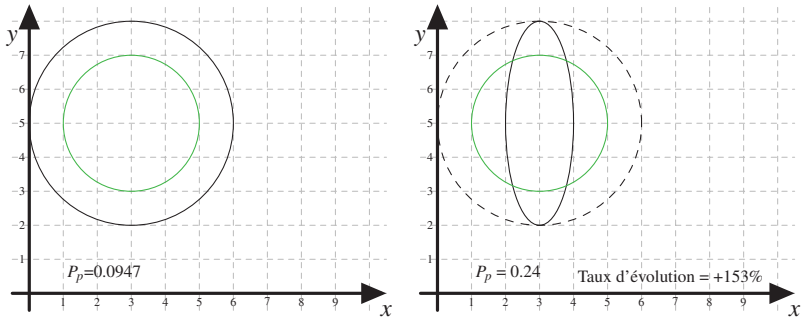


(c) Amélioration potentiellement apportée par le deuxième triplet

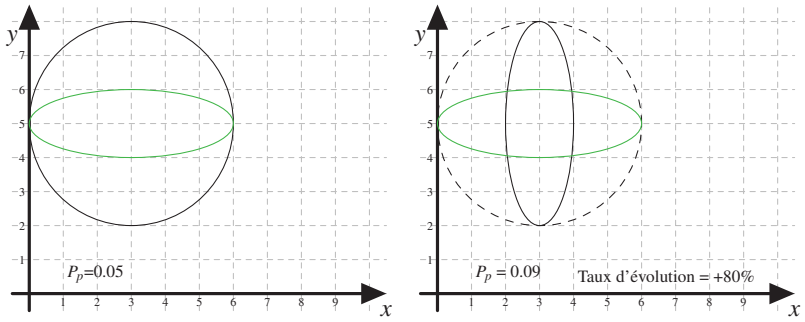
FIGURE IV.2 – Illustration de la différence d'apport vis-à-vis de la précision entre deux triplets. En vert : l'objectif ; En traits pleins et noirs sur la figure de gauche : la covariance initiale. En traits pleins et noirs sur les figures de droite : la covariance estimée après détection.

Exemple 2 : influence de l'objectif sur le choix des triplets

Précédemment, la matrice objective $\mathbf{C}_I$ a été présentée. Sa formalisation permet d'avoir des attentes de précision différentes suivant les coordonnées. Cela permet de gérer des grandeurs hétérogènes comme l'orientation et la position ou encore de privilégier certaines directions comme x ou y.



(a) Objectif de précision identique dans les deux directions x et y



(b) Objectif de précision privilégiant la direction y

FIGURE IV.3 – Influence de l'objectif sur l'importance des triplets. En vert : l'objectif, en traits pleins sur la figure de gauche la covariance initiale, en traits pleins sur la figure de droite la covariance estimée après détection, la mise à jour est la même dans les deux situations mais dans le premier cas, une plus grande partie de l'objectif est atteinte par rapport au second.

La figure IV.3 illustre ce processus. Deux types d'objectifs sont présentés, le premier (figure IV.3a) est homogène dans toutes les directions et est donc représenté par un cercle, le second (figure IV.3b) privilégie la direction y et est donc représenté par une ellipse. La détection réalisée est la même dans les deux situations. Les valeurs de l'objectif de précision P_p sont calculées dans les deux cas. Dans le deuxième cas, ce terme est moins élevé que dans le premier alors que les mises à jour suite à une détection seraient rigoureusement les mêmes. Cet exemple permet d'illustrer comment le choix de l'objectif peut influer sur la pertinence d'un triplet.

IV.2.2 Objectif d'intégrité

Nous venons de voir que nous désirions atteindre une certaine précision pour la localisation, cependant un tel objectif n'est pas suffisant. Comme cela a déjà été souligné, il faut avoir une *croyance vraie et justifiée*, cela signifie que le système doit être capable

de justifier son estimation, de dire si elle est juste ou fausse. [52] fait remarquer que pour beaucoup d'applications, il est extrêmement important de « *savoir quand nous ne savons pas* » et que cela doit faire partie des critères de classification des algorithmes. Le système doit donc être capable de faire une introspection en temps-réel. Cette manière de faire possède plusieurs avantages :

- elle peut permettre au robot de ne pas chercher à exécuter certaines tâches quand il n'est pas sûr de lui ;
- elle lui donne l'opportunité de chercher les causes des erreurs (capteurs défectueux, carte erronée, ...) ;
- elle peut également lui permettre de chercher à réparer ses erreurs comme nous le verrons dans le chapitre VI.

Dans la méthode présentée ici, la croyance est caractérisée par le terme d'intégrité qui peut se définir ainsi : la position estimée est intègre quand la position réelle du robot appartient à la zone d'incertitude estimée autour de la position estimée. La confiance en l'estimation sera notée P_i et représentera la probabilité d'être effectivement intègre. Pour calculer la confiance P_i de l'estimation, il est nécessaire de prendre beaucoup d'événements en compte comme par exemple :

- les caractéristiques des capteurs et des détecteurs ;
- l'occultation possible des amers ;
- les risques de mauvaise association ;
- etc.

Ce calcul n'est pas aisé et est réalisé à l'aide d'un réseau Bayésien que nous décrivons dans le chapitre V. Nous nous contentons ici de le prendre pour acquis.

IV.2.3 Bilan des objectifs

Les deux objectifs fixés sont tout aussi importants. Par exemple, si le robot sait qu'il est sur la Terre avec une confiance de 1 c'est-à-dire qu'il est absolument certain de cette information, cela sera sans doute vrai mais certainement pas suffisant car l'imprécision sera trop importante. Inversement, si le robot pense savoir où il se trouve avec une précision de l'ordre du millimètre mais avec une confiance de 1%, cela sera tout aussi inutile.

Finalement l'objectif sera double : la précision et la confiance. Le système développé doit donc être capable d'atteindre cet objectif de manière performante. Notre processus est un processus Top-Down, c'est-à-dire que les données capteur vont être sélectionnées selon un critère dépendant de l'objectif. Dans le contexte d'un objectif multiple, la définition de ce critère n'est pas aisée. Dans [142], le critère est défini comme étant une pondération selon les différents objectifs fixés. Dans notre cas il serait de la forme $\lambda P_i + \psi P_p$. Ce type de critère oblige à donner un poids à chacun des objectifs, ce qui revient à assigner une importance plus ou moins grande à chacun des objectifs. Ces poids sont souvent définis de manière arbitraire, nous privilégions donc ici une approche évitant au maximum cette paramétrisation.

IV.3 Sélection du triplet

Les objectifs à atteindre étant définis, il faut maintenant sélectionner à chaque instant le triplet perceptif qui permette d'atteindre au mieux ces objectifs. Dans un premier temps, un exemple est présenté qui permet d'appréhender de façon intuitive le processus de sélection du meilleur triplet, dans un second temps l'approche dévelop-

pée est présentée ici et enfin une application pratique montre la pertinence de notre choix.

IV.3.1 Illustration du processus par un exemple

La figure IV.4 montre une illustration de ce principe : un véhicule est positionné au point *O* avec une certaine incertitude sur cette localisation, représentée par l'ellipse noire. Cela signifie que le véhicule peut se situer n'importe où dans cette zone. Différents amers (points A, B, C, D et E) sont également présents. Les positions de ces différents points sont supposées connues et accessibles de façon précise via une carte géoréférencée. Le système doit donc détecter ces différents amers, les associer avec la carte et mettre à jour l'estimation de sa position afin de préciser sa localisation.

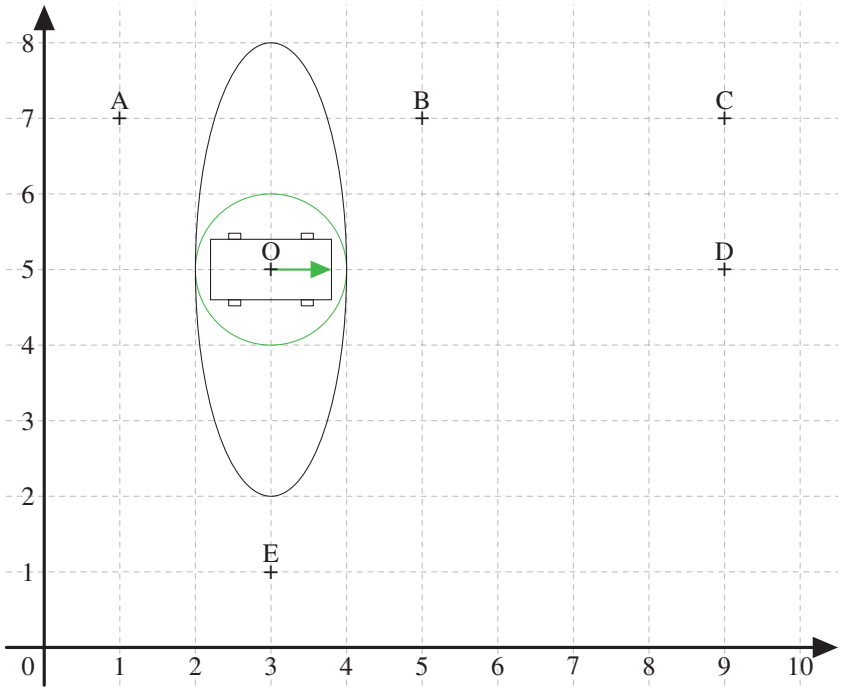


FIGURE IV.4 – Illustration du principe de sélection. L'ellipse noire représente l'incertitude autour de la position estimée *O* du véhicule. Le cercle vert représente l'objectif de précision fixé. Des amers sont situés aux points A, B, C, D et E.

Dans cet exemple, le véhicule est orienté vers la droite et possède un capteur lui permettant de percevoir ce qui se trouve devant lui avec un angle de vue de -90 à +90 degrés. Pour des raisons de clarté et de simplification, le véhicule dans cet exemple possède seulement un capteur et un détecteur. Le détecteur permet, à partir des données du capteur, de connaître la distance approximative entre le véhicule et l'amer. Les différents triplets perceptifs se distingueront donc entre eux seulement par les amers. Pour cet exemple, le triplet sera uniquement aparenté à un amer mais l'extension multi-capteur, multi-détecteur suivra le même principe.

L'objectif de précision est donné par la matrice $\mathbf{C}_I$:

$$\mathbf{C}_I = \begin{pmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{yx} & \sigma_y^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Seules l'ordonnée et l'abscisse de la position du véhicule (en mètres) ont été prises en compte dans cet exemple didactique. Il faut bien noter que dans la réalité, l'orientation est également prise en compte et se traduirait ici par l'ajout d'une ligne et d'une colonne supplémentaires. L'objectif est représenté par un cercle vert sur la figure IV.4. Afin de ne pas surcharger les figures IV.5a, IV.5b, IV.5c, IV.5d,et IV.5e, il n'est pas à nouveau représenté mais il reste identique.

Chaque amer possède différentes propriétés selon l'état actuel du robot :

- le point *A* n'est jamais visible puisque le capteur ne peut observer que selon un angle allant de -90 degrés à +90 degrés. Il n'est donc pas utile de chercher à le détecter ;
- le point *B* est observable où que soit le véhicule dans la zone d'incertitude ;
- les points *C* et *D* sont observables mais sont voisins, il y a donc un risque de les confondre ;
- le point *E* est celui qui améliore le mieux la position du véhicule mais il ne peut être visible que si ce dernier se situe dans la partie gauche de la zone d'incertitude.

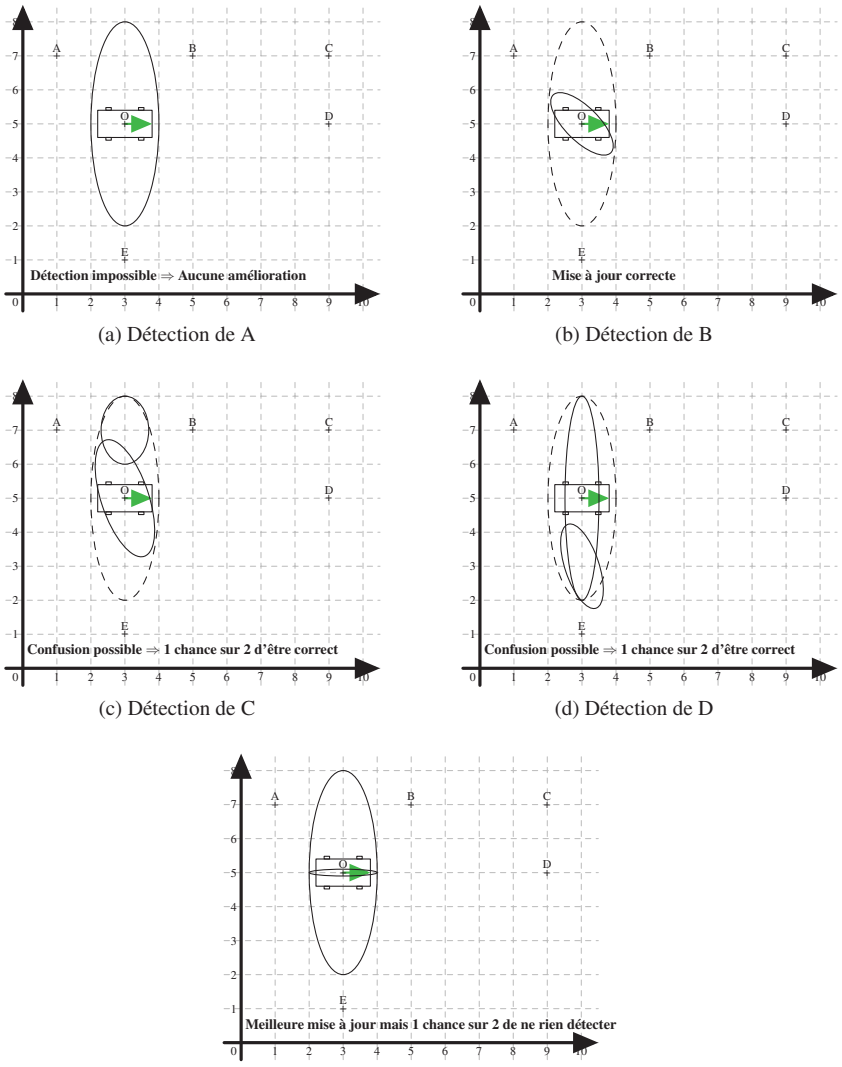
La figure IV.5 illustre les différentes mises à jour possibles après les détections de chaque amer. Les figures IV.5c et IV.5d montrent deux types possibles de mises à jour car, dans ces deux situations, il y a un risque de détecter un amer autre que celui recherché et donc de faire une mise à jour incorrecte.

Au final, nous pouvons affirmer qu'aucun des différents triplets disponibles n'est parfait mais *B* est le triplet qui offre le meilleur compromis au niveau de la précision apportée et de la confiance (pas de risque d'ambiguïté donc la confiance espérée est plus importante), c'est donc ce dernier qu'il faut choisir. C'est exactement le comportement que nous recherchons pour notre système. En prenant en compte les caractéristiques des capteurs, détecteurs et amers disponibles, le but va être de sélectionner celui qui nous apporte le meilleur compromis, pour atteindre nos objectifs à la fois de précision et d'intégrité.

IV.3.2 Triplets perceptifs disponibles

Une étape préliminaire est requise avant de pouvoir sélectionner le meilleur triplet, la construction des différents triplets disponibles. Comme nous l'avons vu dans la section IV.3.1, il faut trois éléments pour constituer un triplet : un capteur, un détecteur et un amer. Le capteur est disponible sur le véhicule utilisé, le détecteur est une brique logicielle développée pour cette application et l'amer est fourni par notre environnement ou plus précisément par la carte géoréférencée de notre environnement. La carte que nous utilisons ici est une carte géoréférencée sémantique dans laquelle différentes données hétérogènes peuvent être stockées comme des murs, des trottoirs, des marquages au sol, des éléments de signalisation (feux rouges, panneaux de stop, ...), etc. A titre d'exemple, nous montrons ici une carte extraite d'Open Street Map (figure IV.6). Suivant l'endroit où se trouve le robot, le type d'amer détectable peut être totalement différent. A chaque instant, la carte est utilisée pour connaître les différents amers disponibles. Le nombre d'amers impacte directement le nombre de triplets perceptifs. Soient N_a, N_c, N_d le nombre d'amers, de capteurs et de détecteurs, ainsi le nombre maximal de triplets dans le pire des cas est de $N_a \times N_c \times N_d$. Il est important de noter qu'une préselection des amers est effectuée, en effet seuls ceux potentiellement visibles sont récupérés. Il est inutile de prendre comme amer un bâtiment qui se trouve à plusieurs dizaines de kilomètres de notre position.

Cette préselection spatiale doit prendre en compte deux éléments : la zone de visi-



(a) Détection de A

(b) Détection de B

(c) Détection de C

(d) Détection de D

(e) Détection de E

FIGURE IV.5 – Mise à jour possible suivant la sélection des amers. Dans chaque cas, en traits pleins noirs la ou les incertitudes possibles, en hachuré l'incertitude précédente, le point O est la position réelle du véhicule.

bilité du capteur et l'incertitude que nous avons sur la position du véhicule (et donc in fine sur la position du capteur). Le calcul de cette sélection est expliqué dans la partie VII.2.

IV.3.3 Critère de sélection des triplets

Le triplet le plus pertinent est défini comme étant le triplet qui permet d'atteindre le plus sûrement possible nos objectifs de confiance et de précision. Le choix d'un critère pertinent est toujours délicat et de nombreux articles ont été publiés en ce sens notam-

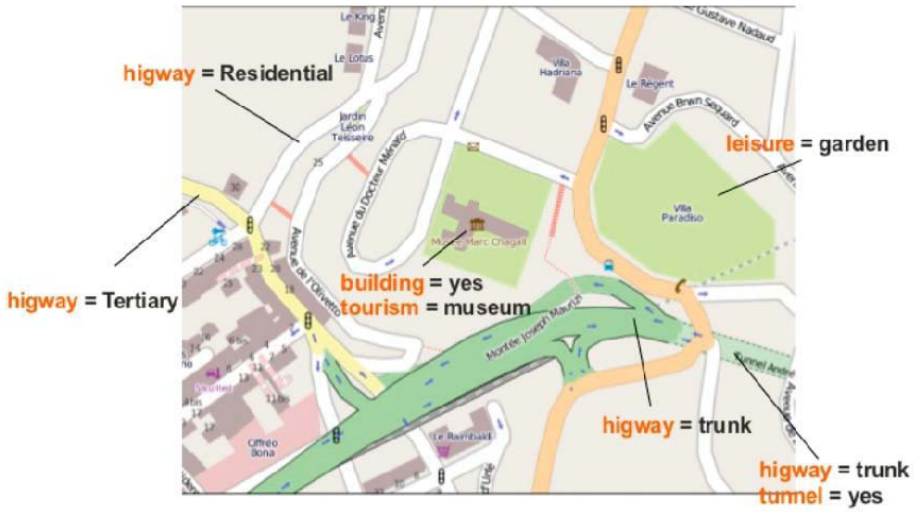


FIGURE IV.6 – Exemple de carte d'Open Street Map. Des amers comme des bâtiments, des tunnels sont représentés et géoréférencés

ment pour tout ce qui concerne la théorie de la décision [44]. Dans notre cas, l'action de sélection consiste en la sélection d'une action de perception. De nombreuses approches de *sensor management* vues dans le chapitre III traitent du problème de sélection du meilleur capteur ou plus exactement de l'ordonnancement optimal des capteurs. Le but est bien évidemment de converger le plus rapidement possible vers l'objectif visé de manière précise et fiable. Dans notre cas, c'est la notion de triplet qui est usitée et pas seulement celle de capteur mais le principe reste similaire. Dans la littérature, deux approches sont décrites : les approches dirigées par la tâche à accomplir, c'est-à-dire par l'objectif, et celles guidées par l'information. Les premières sont souvent délicates à mettre en œuvre dans le cas d'objectif multiples [58]. En effet, avec ces méthodes il y a un risque d'atteindre un minimum local et donc de bloquer le processus.

Pour illustrer ceci, reprenons un des exemples précédents : supposons que nous sachions seulement que nous sommes sur la Terre, cela est très imprécis et n'est pas exploitable. Le système doit donc détecter autre chose dans son environnement pour être plus précis, étant donné l'imprécision de la position il y a de fortes chances pour que l'amer recherché puisse être confondu avec un autre et que la détection donne suite à une mauvaise association. Ceci implique donc que la confiance chute et donc que l'on s'éloigne de l'objectif final en terme de confiance. Si le critère n'est pas bien conçu, le système peut ainsi rester bloqué.

Le second type d'approche consiste à utiliser l'information. Le but n'est plus d'utiliser directement l'objectif comme critère mais l'information. Le triplet sélectionné sera alors le triplet qui maximisera l'information ou plus exactement l'information attendue. En effet, il est possible qu'un triplet apporte beaucoup d'information s'il est détecté mais peu s'il ne l'est pas (la non-détection apporte en effet une information, c'est un point important sur lequel nous reviendrons plus tard). Il faut donc être capable de prendre tout ceci en compte. Pour la détermination de notre critère de sélection, nous avons déterminé quels étaient les différents points importants que notre critère devait respecter pour atteindre au mieux nos objectifs.

Primauté des triplets apportant le maximum d'information vis-à-vis de la confiance

En accord avec les méthodes qui cherchent à maximiser l'information, tout triplet qui n'apporte pas d'information ne doit pas être sélectionné. En parlant d'information apportée par un triplet, nous ne parlons pas d'information au sens général tel qu'il est utilisé par les méthodes de *sensor management* vues auparavant, nous désignons plus particulièrement le fait que la détection d'un triplet apporte de l'information sur la confiance dans la position estimée. Par exemple, si nous détectons un triplet à un instant t, qu'aucun changement ne survient pour le robot, c'est-à-dire entre autres que le robot ne bouge pas, l'environnement est comme figé, détecter derechef ce triplet n'apportera pas de nouvelle information sur la confiance. Au contraire la prise en compte de cette même information peut induire des phénomènes de surconvergence en raison de la forte corrélation des données. Cela est d'autant plus vrai avec un filtre de Kalman classique qui est un estimateur optimiste et qui risque donc de surconverger, donnant ainsi une estimation finalement fausse (certaines méthodes ont cependant été développées pour remédier à ce problème [17, 65] comme l'intersection de covariance [69]). Ce problème d'information corrélée peut causer de véritables soucis en robotique [149]. Ainsi non seulement un triplet n'apportant pas de nouvelle information ne nous aidera pas à atteindre notre objectif mais en plus il risque de nous en éloigner. Bien évidemment, cet événement n'est pas binaire, chaque triplet apporte une information plus ou moins corrélée. Par exemple, si un triplet a été détecté à un moment donné et que le véhicule a effectué un déplacement significatif depuis, il pourra alors à nouveau apporter de l'information. L'événement *L'information apportée par ce triplet à l'instant k est corrélée à l'information précédemment acquise* est noté ici CI_k. Nous devons modéliser cette corrélation par une fonction qui vaut 0 si le triplet n'a jamais été sélectionné auparavant, 1 quand la distance parcourue depuis le dernier essai de détection est nulle et 0 quand cette distance tend vers l'infini. La probabilité que l'information soit corrélée est donnée par une loi de Poisson indicée selon la distance D parcourue par le robot, comme dans la formule suivante :

$$P(CI_k) = \begin{cases} 0 & \text{si le triplet n'a jamais été détecté auparavant} \\ \exp^{-\gamma \times D} & \text{sinon.} \end{cases} \tag{IV.5}$$

Avec :

- D : la distance parcourue depuis la dernière détection de ce triplet ;
- γ : facteur d'oubli (fixé dans notre cas à 0.7 de manière empirique).

Primauté des triplets apportant la meilleure précision

Il est très clair que nous visons ici à atteindre rapidement l'objectif de précision. Pour chaque triplet, nous allons calculer dans quelle mesure l'objectif de précision serait atteint. Cela signifie que le terme P_p, défini par l'équation IV.3, est évalué pour chaque triplet, plus la valeur obtenue est élevée, plus la détection du triplet permettra de se rapprocher de la précision souhaitée.

Primauté des triplets maximisant la probabilité d'être détecté

Jusqu'à présent, le bon triplet est considéré comme toujours détecté mais ce n'est évidemment pas toujours le cas. En effet, un système de sélection serait idéal si le détecteur était parfait (taux de faux positifs et taux de faux négatifs nuls) mais ce n'est pas le cas. Le critère de sélection doit tenir compte de cette non-idéalité.

Notre système de sélection serait idéal si deux phénomènes se présentaient : d'une part que le détecteur renvoie effectivement quelque chose et d'autre part que le résultat du détecteur corresponde bien à la signature de l'amer recherché. Pour cela, l'événement caractérisant la bonne adéquation entre l'amer recherché et la détection est noté *db*. Il faut donc choisir le triplet qui maximise la probabilité de cet événement. Cet événement est directement relié à la confiance. En effet, si la probabilité de réaliser une bonne détection est forte, alors après détection l'association de données est très probablement juste donc la probabilité d'être intègre sera importante. La probabilité d'une bonne détection n'est pas évidente à calculer car elle dépend des caractéristiques du détecteur, du capteur, de l'amer, de l'environnement (phénomène d'occultation, voisins similaires, ...).

L'interaction de tous ces événements sera prise en compte par un Réseau Bayésien que nous présentons dans le chapitre V.

L'objectif est donc de définir un critère de sélection qui présente le meilleur compromis pour prendre en compte les différentes contraintes listées ci-dessus.

Ainsi l'équation suivante a été utilisée pour définir notre critère de sélection :

$$\boxed{C = P_p \times P(db) \times P(\overline{CI_k})} \tag{IV.6}$$

(IV.7)

Pour chaque triplet disponible T_i, le critère correspondant $C(T_i)$ sera calculé. Le meilleur triplet T_b sera alors celui qui aura le critère le plus élevé :

$$T_b = \arg\max_{T_i}(C(T_i)) \tag{IV.8}$$

IV.4 Principe de focalisation et détection

A ce stade, nous supposons que le meilleur triplet perceptif $T_b = \{L_b, S_b, D_b\}$ a été sélectionné grâce à l'utilisation du critère vu précédemment. La prochaine étape est donc d'essayer de détecter effectivement l'amer L_b avec le détecteur D_b et le capteur S_b correspondants. Cette détection est facilitée par le fait que nous savons ce que nous recherchons et que nous avons une estimation de l'endroit où nous pourrions trouver ce que nous recherchons. Par exemple, si l'amer que nous recherchons est une ligne blanche caractérisant un bord de voie de circulation à partir d'une caméra, la zone de recherche de cet amer doit se limiter au maximum à la moitié inférieure de l'image. Le détecteur, quels que soient l'amer et le capteur utilisés, peut être orienté ou focalisé afin d'atteindre plus facilement la détection souhaitée. Cette façon de faire fait partie des méthodes de détection actives [12]. Nous faisons ici une double focalisation : une focalisation spatiale et une focalisation dans l'espace des paramètres.

Focalisation spatiale

La focalisation spatiale consiste à déterminer une zone d'intérêt dans l'image capteur. Cette zone d'intérêt caractérise l'emplacement où rechercher la signature de l'amer sélectionné. Cette zone d'intérêt est définie en prenant en compte la position de l'amer dans la carte, la position estimée du véhicule, la position estimée du capteur et les incertitudes liées à ces estimations. Ainsi plus notre incertitude sur la position du véhicule est faible, plus nous pouvons être précis dans la recherche d'amers. Ici, l'incertitude

des amers est fixe puisqu'elle est fournie par une carte que nous ne remettons pas en cause.

Ce processus est illustré sur la figure IV.7.

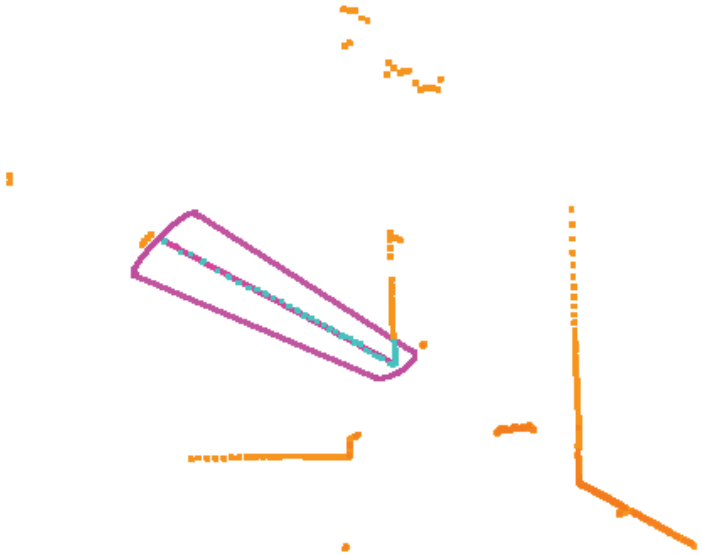


FIGURE IV.7 – Focalisation spatiale sur des données issues d'un télémètres laser. Les données fournies par le télémètre laser sont en orange, la zone de focalisation est en magenta, les données dans la région d'intérêt sont en bleu.

Ce processus de focalisation permet d'une part d'améliorer le rapport Signal sur Bruit et d'autre part de diminuer le temps de calcul en diminuant la quantité de données à traiter.

Focalisation dans l'espace des paramètres

La focalisation spatiale permet de ne pas rechercher notre amer dans toute l'image capteur. Dans ce processus de focalisation spatiale toutes les caractéristiques de l'amer n'ont pas été utilisées. En effet, pour le moment, seule la position estimée de l'amer a été prise en compte. Par exemple, si nous recherchons un poteau dans l'espace télémètre cela signifie que nous recherchons la moitié d'un cercle de rayon r_a propre à l'amer. Si nous détectons un demi-cercle dont le rayon estimé n'est pas compatible avec l'amer réel alors nous pouvons en déduire que le détecteur n'a pas renvoyé le bon résultat. La focalisation dans l'espace des paramètres permet ainsi d'éliminer la majorité des faux positifs.

Cette double focalisation permet ainsi de réduire notablement les erreurs de détection. De plus, si l'estimation de la pose de notre véhicule devient de plus en plus précise, les caractéristiques de focalisation le seront tout autant et la probabilité d'avoir des fausses détections s'en trouvera drastiquement réduite.

IV.5 Application pratique : évolution du critère dans un cas statique

La stratégie de sélection du triplet, en utilisant le critère défini par l'équation IV.6, est appliquée dans notre système. Nous présentons ici une mise en situation où nous voyons comment le critère est utilisé et l'évolution des différents termes au fur et à mesure que le processus se poursuit.

IV.5.1 Présentation de la situation

Cet exemple (figures IV.8 et IV.9) a été créé avec l'environnement de simulation réaliste Cobaye développé par l'entreprise 4d-virtualiz [99] (voir section VII.2). Nous disposons ici d'un environnement constitué de 8 murs, certains visibles depuis le véhicule, d'autres non et d'autres de manière seulement partielle. Le véhicule est équipé d'un capteur unique, d'un télémètre, et d'un détecteur de murs. Pour l'aspect didactique de notre exemple, la position du capteur est confondue avec celle du véhicule. A chaque instant, le critère de sélection est calculé pour chaque triplet. Afin d'illustrer au mieux les différents phénomènes, la scène est considérée comme statique et le véhicule comme immobile. L'objectif de précision est défini par la matrice de variance-covariance $\mathbf{C}_I$:

$$\mathbf{C}_I = \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{yx} & \sigma_y^2 & \sigma_{y\theta} \\ \sigma_{\theta x} & \sigma_{\theta y} & \sigma_\theta^2 \end{pmatrix} = \begin{pmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}$$

L'objectif est donc d'avoir un écart-type de $\sqrt{0.2} = 0.447$ mètres d'incertitude en abscisse et en ordonnée, et de 0.1 radians (soit 5.73 degrés) d'incertitude sur l'orientation.

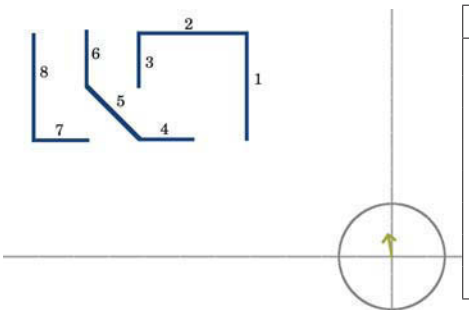
IV.5.2 Évolution de l'algorithme

Étape 1

Initialement (figure IV.8a), l'incertitude est identique en x et en y et est donc représentée par un cercle. L'incertitude sur l'angle θ est très faible (de l'ordre de 1°). L'incertitude de départ étant un cercle, comme pour l'objectif de précision, aucune direction n'est privilégiée et donc l'apport en précision P_p est le même pour tous les triplets. Aucun triplet n'a été sélectionné avant puisque nous sommes à la première étape, cela implique qu'ils apportent tous la même information et que la probabilité $P(CI_k)$ d'avoir une information corrélée est de 0. Le triplet qui sera sélectionné sera donc celui qui aura la meilleure probabilité de détection $P(db)$. Tous les résultats des calculs sont donnés sur le tableau de la figure IV.8a. Ceci prend en compte le phénomène d'occultation (le Mur 2 est totalement occulté par le Mur 1), la confusion possible entre les amers (le Mur 4 peut être confondu avec le Mur 7), etc. Finalement c'est le Mur 1 qui présente le meilleur compromis entre toutes les caractéristiques prises en compte par le critère. C'est donc cet amer qui est sélectionné et qui est recherché.

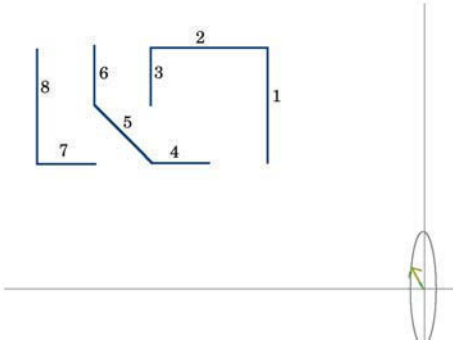
Étape 2

A ce stade, le triplet comportant le Mur 1 a déjà été sélectionné, détecté et utilisé pour la mise à jour de notre position via un filtre de Kalman. Cela implique plusieurs choses : l'incertitude de notre position estimée est plus petite (voir ellipse autour de



	P_p	$P(db)$	$P(CI_k)$	$C(T_i)$
Mur 1	0.24	1	0	0.24
Mur 2	0.24	0	0	0
Mur 3	0.24	0.23	0	0.055
Mur 4	0.24	0.5	0	0.12
Mur 5	0.24	0.09	0	0.02
Mur 6	0.24	0	0	0
Mur 7	0.24	0.37	0	0.088
Mur 8	0.24	0	0	0

(a) Étape 1



	P_p	$P(db)$	$P(CI_k)$	$C(T_i)$
Mur 1	0.27	0.99	0.99	0.0026
Mur 2	0.67	0	0	0
Mur 3	0.27	0.59	0	0.159
Mur 4	0.67	0.49	0	0.328
Mur 5	0.56	0	0	0
Mur 6	0.27	0	0	0
Mur 7	0.67	0.48	0	0.321
Mur 8	0.27	0	0	0

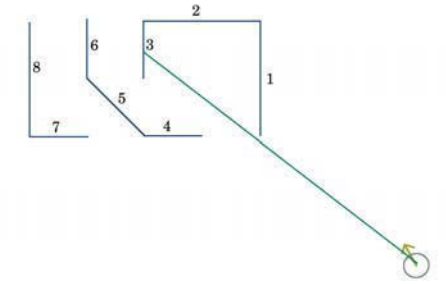
(b) Étape 2

FIGURE IV.8 – Évolution du critère : étapes 1 et 2. P_p quantifie l'aport en précision apporté par le triplet vis-à-vis de l'objectif de précision. $P(db)$ est la probabilité de détecter correctement le triplet recherché. $P(CI_k)$ est la probabilité d'avoir une information corrélée aux précédentes observations. $C(T_i)$ est le critère utilisé pour sélectionner le meilleur triplet perceptif.

la position estimée sur la figure IV.8b). Le Mur 1 ne nous apportera plus d'information ou extrêmement peu puisque le robot n'a pas bougé. L'incertitude sur la position ayant évolué, cela implique que la probabilité de bonne détection de chacun des triplets a également évolué (la zone de focalisation sera plus petite, levant ainsi les ambiguïtés potentielles ; certains amers, qui étaient probablement occultés, voient cette probabilité fluctuer également). Les nouvelles probabilités sont recalculées grâce au réseau Bayésien décrit dans le chapitre suivant et données dans le tableau présenté sur la figure IV.8b. Sans surprise, la probabilité de ne pas apporter de nouvelle information $P(\overline{CI_k})$ passe de 0 à presque 1 pour le Mur 1. La détection précédente a privilégié un axe particulier pour l'incertitude, qui n'est donc plus anisotrope. Suivant l'orientation des murs la précision apportée par chacun d'entre eux vis-à-vis de l'objectif n'est pas la même. La direction privilégiée pour la détection suivante est celle qui est perpendiculaire au Mur 1, ce qui se voit sur les Murs $2,4,7$. Au contraire, les murs $1,3,6,8$,qui sont parallèles au Mur 1, sont ceux qui apportent le moins de précision, le Mur 5 se situant entre les deux. Les probabilités de bonne détection sont également mises à jour.

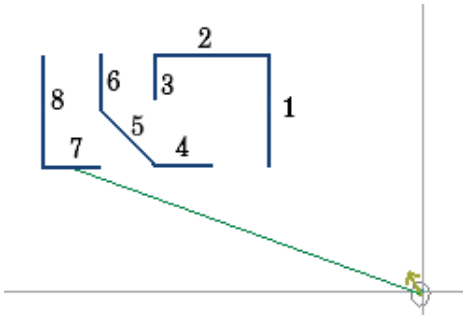
Finalement c'est le Mur 4 qui offre le meilleur compromis et qui va donc être choisi.

Étapes 3 et 4



	P_p	$P(db)$	$P(\overline{CI_k})$	$C(T_i)$
Mur 1	0.73	0.99	0.99	0.007
Mur 2	0.73	0	0	0
Mur 3	0.73	0.87	0	0.6351
Mur 4	0.73	0.33	0.99	0.002
Mur 5	0.73	0.9	0	0.02
Mur 6	0.73	0	0	0
Mur 7	0.73	0.48	0	0.35
Mur 8	0.24	0	0	0

(a) Étape 3



	P_p	$P(db)$	$P(\overline{CI_k})$	$C(T_i)$
Mur 1	0.75	0.99	0.99	
Mur 2	0.79	0	0	0
Mur 3	0.75	0.89	0.99	0.0067
Mur 4	0.79	0.49	0.99	0.003
Mur 5	0.77	0	0	0
Mur 6	0.75	0	0	0
Mur 7	0.79	0.48	0	0.37
Mur 8	0.75	0	0	0

(b) Étape 4

FIGURE IV.9 – Évolution du critère : étapes 3 et 4

Après cette seconde étape, la sélection des autres triplets continue selon le même principe. Les étapes 3 et 4 sont données ici pour bien montrer comment le système présenté permet de sélectionner au fur et à mesure les bons triplets afin d'atteindre le plus rapidement possible l'objectif fixé. La réduction de la zone de focalisation permet de lever certaines ambiguïtés, par exemple le Mur 3 (figure IV.9a) peut être détecté sans risque de le confondre avec le Mur 6 ce qui n'était pas le cas à la première étape. Le sélectionner maintenant est donc devenu un choix judicieux ce qui aurait été faux au départ.

IV.6 Conclusion

A l'issue de cette partie, un système de sélection de la meilleure action à réaliser a été présenté. Plusieurs concepts de notre approche ont été présentés et validés sur un exemple précis :

- choix d'une méthode Top-Down ;
- définition d'un critère de sélection d'un triplet pour atteindre un objectif fixé avec prise en compte de l'environnement, capteurs, détecteurs, etc ;
- processus de focalisation rendu possible par le choix d'une stratégie Top-Down.

Tout ce processus s'appuie sur la prise en compte des liens de causalité entre les différents événements modélisés grâce à un réseau Bayésien. Ce réseau est présenté dans le chapitre qui suit.

CHAPITRE V

Modélisation des interactions entre les différents événements avec un réseau bayésien

Sommaire

V.1 Introduction

Dans le chapitre III, le principe des réseaux bayésiens a été présenté. Il a été montré que leur utilisation était pertinente dans la situation qui nous préoccupe. Un réseau bayésien va pouvoir être utilisé à la fois pour calculer des probabilités a priori et a posteriori en prenant en compte les observations de l'environnement par utilisation d'inférence. Par exemple quand les détections successives des triplets réussissent, ces événements sont pris en compte dans le réseau et la probabilité que le véhicule se trouve à la position estimée devient de plus en plus forte. Dans notre cas, le réseau bayésien (figure V.1) déployé va notamment servir à deux niveaux :

- sélection du triplet le plus pertinent (calcul de la probabilité $P(d_b)$ de bonne détection) ;
- mise à jour de la confiance (calcul de la probabilité $P(O_k)$ d'avoir une estimation de la position correcte).

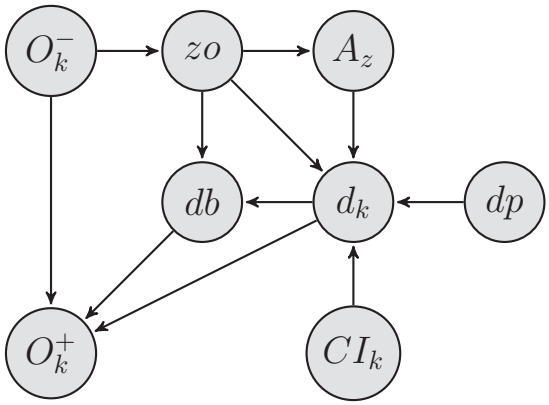


FIGURE V.1 – Réseau bayésien mis en place

Les différents événements présents dans ce réseau vont être présentés par la suite mais il est déjà possible de voir certains termes maintenant familiers, en particulier :

- db : la probabilité de cet événement rentre directement en compte dans le calcul du critère de sélection défini au chapitre précédent. Il s'agit de la probabilité de réaliser une bonne détection. Cette probabilité est cachée mais sera estimée par inférence dans le réseau.
- CI_k : la probabilité de cet événement est également directement utilisée dans le calcul du critère de sélection. Il s'agit de la probabilité que l'information soit corrélée ou non avec les observations précédentes. Dans le chapitre précédent, le calcul de cette probabilité a été expliqué et est indépendant du réseau. Toutefois, cet événement intervient dans le réseau final, non comme une sortie mais comme une entrée du système.

De but en blanc, la compréhension de ce réseau peut être assez délicate. C'est pourquoi dans un premier temps, la construction pas-à-pas du réseau bayésien dans un cadre statique va être effectuée. Dans un deuxième temps, l'aspect dynamique du processus sera alors pris en compte.

V.2 Construction de notre réseau bayésien étape par étape

Le but de cette partie est de montrer comment notre réseau bayésien a été construit pas-à-pas en partant d'un cas idéal où tout est parfait puis en dégradant progressivement ce cas de manière à modéliser au mieux la réalité et ainsi atteindre le formalisme général présenté dans le paragraphe précédent. L'avantage de présenter notre réseau bayésien de cette façon est de mieux comprendre le rôle de chaque nœud, quels événements il modélise et comment il interagit avec les autres nœuds. Pour chaque étape, le réseau bayésien sera présenté, ainsi que les tables de probabilités conditionnelles concernées. Dans ces dernières, les probabilités qui auront été modifiées ou rajoutées seront en rouge. Les nœuds concernés, soit parce qu'ils auront été rajoutés soit parce que leur table de probabilité aura évolué, seront en bleu.

Dans un premier temps, un monde idéal est modélisé. Concrètement, un certain nombre d'hypothèses simplificatrices modélisant un monde idéal sont posées lors de la première étape. Au fur et à mesure des étapes chaque hypothèse sera remise en cause et l'événement associé sera amélioré (définition et modélisation), ceci afin de permettre une bonne compréhension des différents phénomènes, leurs interactions et leur prise en charge. Afin de bien suivre l'évolution du processus, toutes les hypothèses seront listées au début de chaque étape. Sur cette liste, une hypothèse remise en cause dans les étapes précédentes sera barrée tandis qu'une hypothèse en cours de traitement sera en gras et barrée. Cette hypothèse sera alors expliquée, puis remise en cause afin de prendre en compte de manière plus adéquate la réalité. Les tables de probabilités conditionnelles sont alors données. ϕ dans une ligne d'un tableau signifie que la valeur peut être 0 ou 1, cela est sans importance, dans la réalité cette ligne correspondra à des situations impossibles.

V.2.1 Un monde parfait

Liste des hypothèses :

- les détecteurs sont parfaits ;
- tout amer est observable ;
- il n'y a pas de phénomène d'occultation ;
- si la position estimée n'est pas intègre, l'amer recherché n'est pas observable ;
- il n'y a pas d'ambiguïté entre les amers ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle ;

Les événements utilisés par le réseau bayésien en vue de modéliser les hypothèses posées ci-dessus sont :

O_k^- la localisation a priori à l'instant k est intègre ;

O_k^+ la localisation a posteriori (c'est-à-dire après mise à jour de l'état par la détection réalisée) à l'instant k est intègre ;

dk Le détecteur a renvoyé un résultat positif (une détection a pu être réalisée) ;

db Le détecteur a renvoyé un résultat positif qui correspond à l'amer recherché.

Il est alors possible de définir le réseau bayésien de la figure V.2

Les tables de probabilité correspondant à cette première version sont données dans les tableaux V.1.

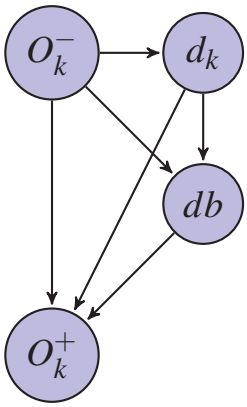


FIGURE V.2 – Réseau bayésien dans un monde parfait

Ok^-	$P(dk)$
0	0
1	1

Ok^-	dk	$P(db)$
0	0	0
0	1	ϕ
1	0	0
1	1	1

Ok^-	dk	db	$P(Ok^+)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	ϕ
1	1	0	0
1	1	1	1

TABLE V.1 – Tables de probabilité de db, dk et Ok^+ dans un cas parfait

Pour la table correspondant à l'événement dk et compte tenu des hypothèses fixées précédemment deux situations peuvent se présenter :

- la localisation initiale n'est pas intègre, alors le détecteur retournera obligatoirement un résultat négatif (1ère ligne de la table) ;
- si la localisation initiale est intègre, alors le détecteur retournera un résultat positif (2ème ligne de la table).

Pour la table associée à l'événement db, si le détecteur renvoie un résultat négatif ($dk = 0$) alors $P(db) = 0$ par définition puisqu'il ne peut y avoir bonne association sans détection. Si le détecteur renvoie un résultat positif ($dk = 1$) alors il est forcément vrai ($P(db) = 1$) puisqu'il est supposé parfait.

Même s'il pourrait être simplifié davantage, ce réseau bayésien composé de 4 nœuds permet une première modélisation de notre processus dans un cas idéal.

V.2.2 Détecteurs réels

Liste des hypothèses :

- ~~**les détecteurs sont parfaits**~~ ;
- tout amer est observable ;
- il n'y a pas de phénomène d'occultation ;
- si la position estimée n'est pas intègre, l'amer recherché n'est pas observable ;
- il n'y a pas d'ambiguïté entre les amers ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle.

Précédemment, nous avons supposé que nos détecteurs étaient parfaits, c'est-à-dire que si l'amer se trouvait dans la zone de visiblité il était détecté sinon le détecteur ne renvoyait rien. Dans la réalité nos détecteurs sont loin d'être parfaits. La qualité du détecteur ou de la détection peut dépendre de plusieurs paramètres liés au capteur, à l'amer ou à l'environnement comme la distance à l'objet, la présence de bruit ou non dans les mesures, ... Afin de prendre en compte toutes ces caractéristiques difficilement quantifiables individuellement, nous avons modélisé ces phénomènes par deux probabilités :

- α probabilité de ne pas détecter alors que l'amer est présent ;
- β probabilité de détecter alors qu'il n'y a pas d'amer (détection d'outlier).

Ces termes proviennent du domaine du traitement de signal statistique où ils sont souvent rencontrés sous les termes de risque de première espèce (α) et risque de seconde espèce (β). Il est facile de comprendre que si l'on veut améliorer l'un des deux paramètres (c'est-à-dire le rendre plus petit), l'autre va généralement être dégradé (donc devenir plus grand).

α va être déterminé en prenant en compte le secteur angulaire sous lequel l'amer est observé. β est déterminé et fixé empiriquement de façon indépendante pour chaque type de détecteur. Ces valeurs pourraient tout à fait être optimisées en fonction de la distance entre l'amer et le robot, l'angle sous lequel est vu l'amer, etc.

La levée de l'hypothèse liée à la qualité du détecteur ne modifie pas le réseau initial mais uniquement les interactions entre les événements (modification de la table liée à d_k, voir tableau V.2).

Ok^-	$P(dk)$
0	β
1	$1-\alpha$

TABLE V.2 – Table de probabilité de dk avec de détecteurs non parfaits

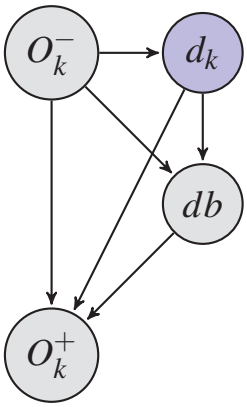


FIGURE V.3 – Réseau bayésien avec des détecteurs non parfaits

On fait également l'hypothèse que si au moins un amer est présent et que le détecteur renvoie une détection, alors c'est effectivement un amer qui a été détecté (et non du bruit). Cela revient à dire que si l'amer est présent, il est suffisamment visible pour empêcher que le détecteur détecte du bruit. Cette hypothèse peut sembler assez forte mais elle est bien vérifiée en pratique et nous la conserverons pour l'ensemble du processus. Le fait que cette hypothèse soit bien vérifiée provient grandement du fait qu'un processus de focalisation est utilisé et que le rapport Signal Sur Bruit se trouve fortement augmenté (voir IV.4).

V.2.3 Observabilité du triplet

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- **~~tout amer est observable~~** ;
- **~~il n'y a pas de phénomène d'occultation~~**
- si la position estimée n'est pas intègre, l'amer recherché n'est pas observable ;
- il n'y a pas d'ambiguïté entre les amers ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle.

Dans cette partie, nous allons lever l'hypothèse sur l'observabilité de l'amer. En effet, plusieurs causes peuvent être à l'origine de la non-observabilité de l'amer :

- distance de vue maximale du capteur ;
- orientation de l'amer vis-à-vis du capteur ;
- occultation par un autre amer ;
- ...

La prise en compte de l'observabilité est caractérisée par l'ajout d'un nouvel événement« l'amer recherché est observable » noté zo dans la structure initiale du réseau (voir figure V.4) et d'une probabilité associée obs (voir tableau V.3).

zo	d_k	$P(db)$
0	0	0
0	1	0
1	0	0
1	1	1

zo	$P(dk)$
0	β
1	$1-\alpha$

O_k^-	$P(zo)$
0	0
1	*obs*

TABLE V.3 – Table de probabilités suite à la prise en compte de l'observabilité

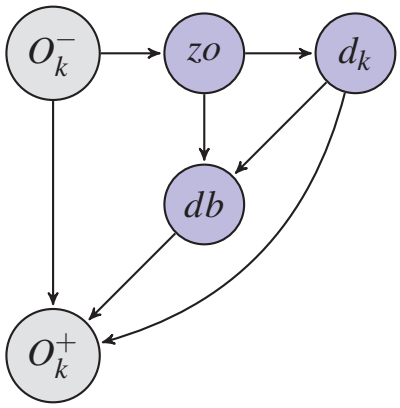


FIGURE V.4 – Réseau bayésien impacté suite à la prise en compte de l'observabilité

Le calcul spécifique de la probabilité *obs* sera décrit dans la section V.3.1.

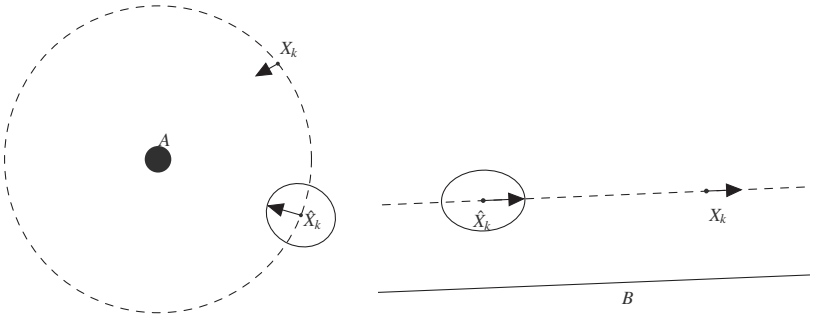
V.2.4 Observabilité de l'amer dans le cas de non-intégrité

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- ~~tout amer est observable~~ ;
- ~~il n'y a pas de phénomène d'occultation~~ ;
- ~~**si la position estimée n'est pas intègre,l'amer recherché n'est pas observable**~~ ;
- il n'y a pas d'ambiguïté entre les amers ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle.

Il peut arriver que la position estimée du robot ne soit pas intègre mais que l'amer recherché soit malgré tout observable. Pour illustrer ce principe, la figure V.5 montre deux situations où cela peut se produire. Dans le premier cas (figure V.5a), la position réelle X_k du robot se trouve sur le cercle en pointillé et le robot est orienté en direction de l'amer A, la position estimée $\hat{X}_k$ se trouve également sur le cercle en pointillé et l'orientation du robot est en direction de l'amer A. Dans cette situation, le robot voit l'amer de la même manière dans une situation ou l'autre. La position estimée n'est pas intègre car la position réelle X_k n'est pas dans la zone d'incertitude autour de $\hat{X}_k$. Malgré la non-intégrité de la position, l'amer est observable. Le cas de figure est similaire dans le deuxième cas (figure V.5b), si le robot se trouve sur la droite en pointillé parallèle au mur et orienté dans la direction de la droite, le mur sera observable de manière identique dans toutes ces situations même si la position estimée n'est pas intègre.

La structure du réseau n'est pas modifiée pour lever cette hypothèse. Seule la première ligne de la table de probabilités conditionnelles liée à zo est modifiée. Cette probabilité est difficilement quantifiable et nous la fixons arbitrairement à 1% car cette situation n'arrive que très rarement.



(a) Observabilité d'un poteau noté A (b) Observabilité d'un mur noté B

FIGURE V.5 – Observabilité de l'amer recherché dans le cas de non-intégrité

O_k^+	$P(zo)$
0	0.01
0	*obs*

TABLE V.4 – Table de probabilité de *zo* avec un amer observable mais une position non intègre

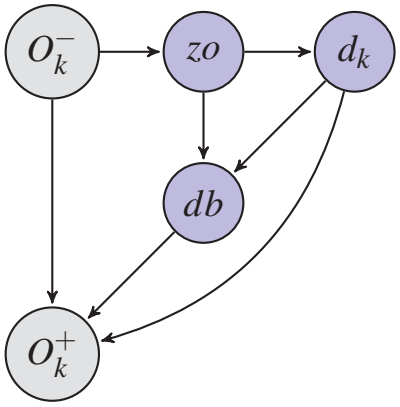


FIGURE V.6 – Réseau bayésien impacté suite à la prise en compte de l'observabilité dans le cas de non-intégrité

V.2.5 Ambiguïté possible entre deux triplets

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- ~~tout amer est observable~~ ;
- ~~il n'y a pas de phénomène d'occultation~~ ;
- ~~si la position estimée n'est pas intègre, l'amer recherché n'est pas observable~~ ;
- **~~il n'y a pas d'ambiguïté entre les amers (1)~~** ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle.

La notion d'ambiguïté se caractérise ici comme le fait que plusieurs amers de même nature puissent se trouver dans la zone de recherche comme par exemple plusieurs

arbres. Dans ce cas, le risque de mauvaise association est plus élevé. Afin de prendre en compte ce phénomène, le nombre d'amers potentiel dans la zone analysée est calculé. Ces amers sont appelés ici des voisins. Les modifications engendrées par la prise en compte de ces ambiguïtés sont doubles.

D'une part dans la table des probabilités de *db* où la probabilité que l'amer détecté soit celui recherché devient 1/(Nombre d'amers compatibles). D'autre part, la valeur de la probabilité α est également modifiée afin de tenir compte du fait que plus il y a d'amers dans la zone plus le détecteur a de chances de renvoyer un résultat.

z	d_k	$P(db)$
0	0	0
0	1	0
1	0	0
1	1	$\frac{1}{nbVoisins+1}$

Ok	$P(dk)$
0	β
1	$1-\alpha^{(nbVoisins+1)}$

TABLE V.5 – Table de probabilités de *db* et *dk* suite à la prise en compte des voisins

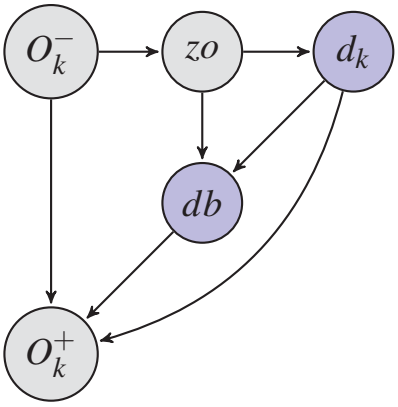


FIGURE V.7 – Réseau bayésien suite à la prise en compte des voisins

Le calcul spécifique du nombre de voisins sera décrit dans la section V.3.2.

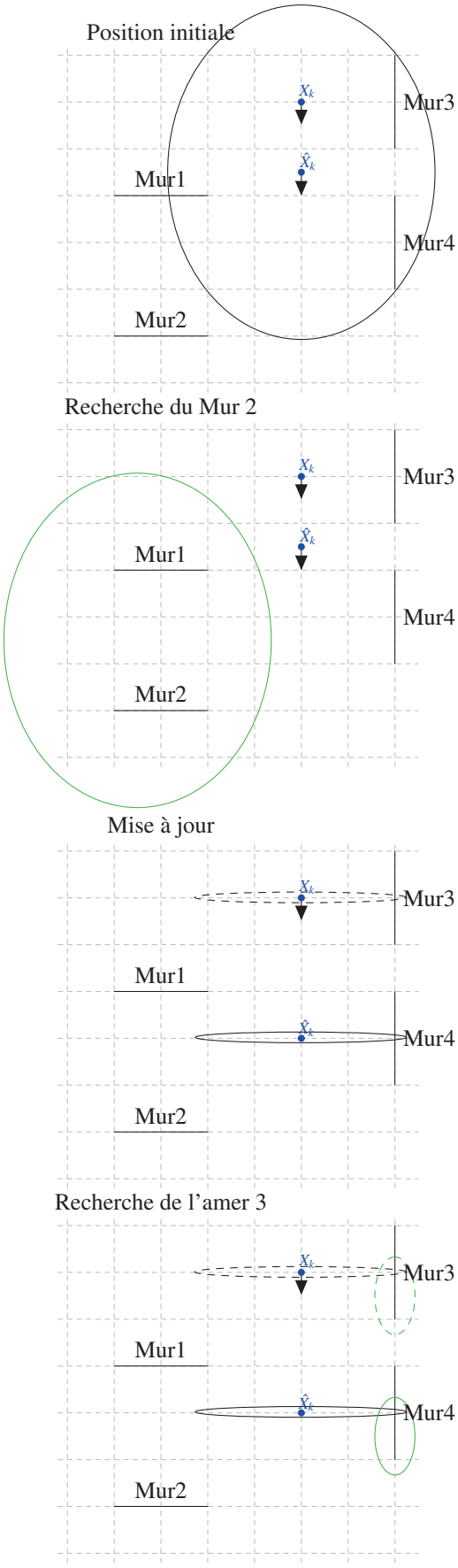
V.2.6 Ambiguïté possible due aux différentes configurations

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- ~~tout amer est observable~~ ;
- ~~il n'y a pas de phénomène d'occultation~~ ;
- ~~si la position estimée n'est pas intègre, l'amer recherché n'est pas observable~~ ;
- **~~il n'y a pas d'ambiguïté entre les amers (2)~~** ;
- si la position estimée n'est pas intègre, il n'y a pas d'amers compatibles observables ;
- chaque triplet apporte une information totalement nouvelle.

Dans la réalité, le nombre de voisins tel que nous l'avons calculé avant ne dépend pas seulement de l'état estimé du véhicule mais également du fait que nous ayons fait ou non une bonne association précédemment.

Pour comprendre cela, la figure V.8 donne un exemple de ce genre de situation.



Au départ, la position estimée ($\hat{X}_k$) est intègre. L'ellipse d'incertitude englobe la position réelle. Le placement relatif du Mur 3 par rapport au Mur 1 est identique au placement relaitf du Mur 4 par rapport au Mur 2, on dit qu'il y a une configuration identique.

Le système sélectionne alors le Mur 2 à détecter. L'ellipse d'incertitude autour de la position est projetée dans l'espace de recherche (ellipse en vert) afin de déterminer la zone de focalisation. Il n'y a pas que le Mur2 dans cette zone, il y a également le Mur 1 qui possède des caractéristiques similaires. C'est ce Mur qui est détecté en lieu et place du Mur 2. Etant donné qu'il y avait un risque de se tromper (1 voisin), la probabilité d'avoir détecté le bon et donc d'avoir une position intègre est au maximum de $1/2$

La mise à jour est alors faussée. Au lieu d'avoir une position estimée autour de X_k, cette dernière se trouve plus en avant.

Le système cherche alors à détecter le Mur 4. Pour cela, il projette son incertitude dans la zone où il croit le rechercher (ellipse verte en trait plein). En réalité, la véritable zone de focalisation est celle de l'ellipse verte hachurée à l'intérieur de laquelle se trouve le Mur 3. La probabilité de se trouver dans la bonne situation est donc toujours de $1/2$. Pour prendre ce phénomène en compte, il faut pouvoir calculer les voisins artificiels dûs à la configuration.

FIGURE V.8 – Exemple de configurations identiques et des impacts sur la localisation

La prise en compte de ce phénomène appelé ici : « ambiguïté suite à des configurations identiques » impacte seulement la table de probabilité *db* (tableau V.6). La structure du réseau reste inchangée (figure V.9).

zo	d_k	$P(db)$
0	0	0
0	1	0
1	0	0
1	1	$\frac{1}{nbVoisins + nbVoisinsConfig + 1}$

TABLE V.6 – Table de probabilité de *db* suite à la prise en compte des configurations similaires

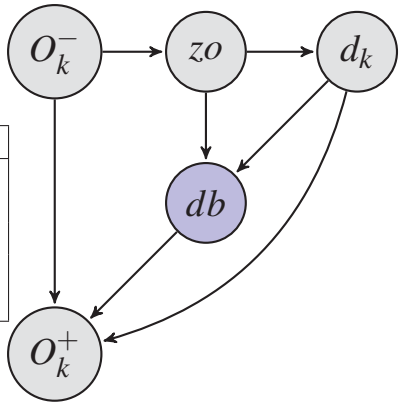


FIGURE V.9 – Réseau bayésien suite à la prise en compte des configurations similaires

Le calcul du nombre de « voisins configurations » est détaillé dans la section V.3.3.

V.2.7 Densité d'amers

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- ~~tout amer est observable~~ ;
- ~~il n'y a pas de phénomène d'occultation~~ ;
- ~~si la position estimée n'est pas intègre, l'amer recherché n'est pas observable~~ ;
- ~~il n'y a pas d'ambiguïté entre les amers~~ ;
- ~~**si la position estimée n'est pas intègre et que l'amer n'est pas observable, il n'y a pas d'amers compatibles observables**~~ ;
- chaque triplet apporte une information totalement nouvelle.

Jusqu'à maintenant, l'hypothèse suivante a été maintenue : *Si la position estimée n'est pas intègre et que l'amer n'est pas observable, il n'y a pas d'amers compatibles observables*. Cette hypothèse est évidemment fausse, en effet si on cherche à détecter un arbre précis dans une forêt et que l'on regarde ailleurs il y a un risque important de détecter malgré tout un arbre. Tout dépend en réalité de la densité d'amers dans notre carte. Le but dans ce réseau est de prendre en compte cette densité. Cette situation est semblable avec celle de la prise en compte des voisins. La différence provient du fait que pour pouvoir estimer le nombre de voisins la position estimée est intègre et le système sait donc dans quelle zone de la carte regarder. Dans le cas qui nous concerne ici l'intégrité a été perdue ce qui signifie qu'il est donc impossible de savoir où le système regarde, l'estimation du nombre de voisins sera donc moins précise et ne peut être estimée qu'à partir de données globales de la carte (densité d'amers dans toute la carte) et non de données locales se rapportant à la zone de focalisation.

Afin de prendre en compte cette densité, un nouvel événement A_z est défini de la façon suivante : il y a au moins un amer dans une zone de la taille T_z de notre zone de

focalisation. La densité d'amers au mètre carré dans la carte est notée A. La probabilité de réaliser une détection sachant qu'au moins un amer est visible et que ce n'est pas l'amer recherché est alors définie de façon similaire à la probabilité de détecter si l'amer était observable (voir tableau V.7).

zo	A_z	$P(d_k)$
0	0	β
0	1	$1-\alpha^{A\times T_z}$
1	0	0
1	1	$1-\alpha^{(nbVoisins+1)}$

zo	$P(A_z)$
0	$\min(1, A\times T_z)$
1	1

TABLE V.7 – Tables de probabilité de d_k et A_z en prenant en compte la densité d'amers compatibles

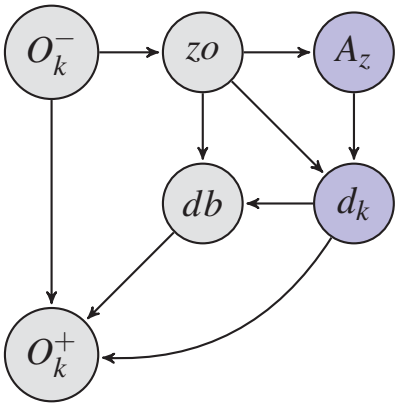


FIGURE V.10 – Réseau bayésien prenant en compte la densité d'amers

V.2.8 Prise en compte de la corrélation éventuelle entre les données

Liste des hypothèses :

- ~~les détecteurs sont parfaits~~ ;
- ~~tout amer est observable~~ ;
- ~~il n'y a pas de phénomène d'occultation~~ ;
- ~~si la position estimée n'est pas intègre, l'amer recherché n'est pas observable~~ ;
- ~~il n'y a pas d'ambiguïté entre les amers~~ ;
- ~~si la position estimée n'est pas intègre et que l'amer n'est pas observable, il n'y a pas d'amers compatibles observables~~ ;
- ~~**chaque triplet apporte une information totalement nouvelle**~~.

Dans la partie traitant du calcul du critère de sélection, le problème de corrélation entre les données d'un même capteur a été abordé. Cette corrélation a été modélisée par un événement noté CI_k. Si un triplet n'apporte pas d'information nouvelle car la donnée est corrélée avec les données déjà fusionnées, il faut être capable de modéliser son impact dans le réseau bayésien. Dans le chapitre précédent, seule la notion de corrélation entre les données avait été prise en compte, ici le concept est mieux modélisé. En effet, le fait de savoir si le système avait réussi ou non la détection permet d'appréhender le résultat d'une future détection. Il est donc nécessaire d'intégrer le passé dans notre sélection d'amers. Nous avons donc ici trois nouvelles situations :

- Le triplet n'a jamais été sélectionné auparavant ;
- Le triplet a été sélectionné auparavant et la détection avait renvoyé un résultat positif ;
- Le triplet a été sélectionné auparavant et la détection avait renvoyé un résultat négatif.

CIk	dp	zo	A_z	$P(d_k)$
0	0	0	0	β
0	0	0	1	$1-\alpha^{A\times T_z}$
0	0	1	0	0
0	0	1	1	$1-\alpha^{(nbVoisins+1)}$
0	1	0	0	β
0	1	0	1	$1-\alpha^{A\times T_z}$
0	1	1	0	0
0	1	1	1	$1-\alpha^{(nbVoisins+1)}$
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

TABLE V.8 – Tables de probabilités conditionnelles de d_k avec prise en compte de la corrélation entre les données

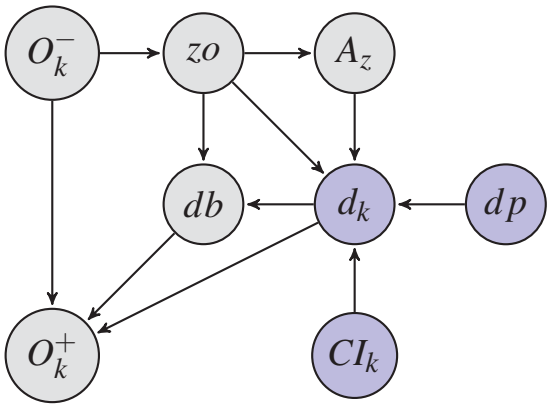


FIGURE V.11 – Réseau bayésien avec prise en compte de l'information apportée par un triplet

Il nous faudrait donc rajouter un événement comportant 3 états distincts dans notre réseau, nous avons fait ici le choix de ne conserver que des nœuds booléens (c'est-à-dire modélisant des événements soit vrais soit faux). C'est pourquoi nous distinguons ici deux événements :

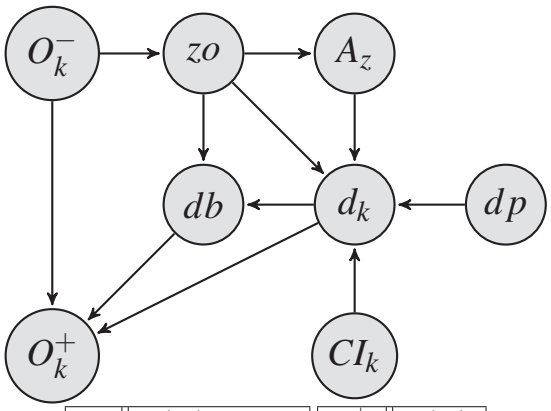
– CI_k l'information apportée par l'amer est corrélée ;
– dp l'amer a été détecté précédemment.

Bien évidemment si l'événement CI_k est faux, le nœud dp ne sera pas pris en compte. Ces deux nouveaux nœuds impactent la détection de l'amer (figure V.11). Le calcul de $P(CI_k)$ a déjà été détaillé dans le chapitre IV. La table de probabilité de d_k est donnée dans le tableau V.8. Quand il y a une nouvelle information ($CI_k = 0$) alors la probabilité de d_k suit les mêmes règles que précédemment. S'il n'y a pas de nouvelle

information (CI_k =1) et que la détection précédente avait réussi, alors la probabilité que celle-ci réussisse est de 1 et inversement.

V.2.9 Récapitulatif

Nous avons maintenant construit notre réseau bayésien en prenant en compte les différents phénomènes énumérés au début. Nous fournissons maintenant ici le réseau bayésien final ainsi que toutes les tables de probabilités conditionnelles à titre informatif sur la figure V.12.



zo	$P(A_z)$
0	$\min(1, A \times T_z)$
1	1

O_k^+	$P(zo)$
0	0.01
0	*obs*

CIk	dp	zo	A_z	$P(d_k)$
0	0	0	0	β
0	0	0	1	$1-\alpha^{A\times T_z}$
0	0	1	0	0
0	0	1	1	$1-\alpha^{(nbVoisins+1)}$
0	1	0	0	β
0	1	0	1	$1-\alpha^{A\times T_z}$
0	1	1	0	0
0	1	1	1	$1-\alpha^{(nbVoisins+1)}$
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

zo	d_k	$P(db)$
0	0	0
0	1	0
1	0	0
1	1	$\frac{1}{nbVoisins + nbVoisinsConfig + 1}$

Ok^-	dk	db	$P(Ok^+)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	ϕ
1	1	0	0
1	1	1	1

FIGURE V.12 – Réseau bayésien complet avec toute les tables de probabilités conditionnelles associées

V.2.10 Évolution

Le réseau bayésien modélisé ici prend en compte un certain nombre d'événements qui sont suffisants dans le cadre de notre application. Cependant, il peut arriver que l'on dispose d'informations sur d'autres événements qui peuvent venir améliorer notre estimation. Il est très facile de rajouter ce genre d'événements car cela n'a qu'un impact limité sur le réseau, en effet seuls les parents directs et les enfants du nœud que l'on désire rajouter sont impactés. A titre d'exemple, nous pourrions rajouter la contrainte que notre véhicule est forcément sur la route et que cela impacte donc la probabilité de voir certains amers ou non. Nous illustrons cela sur la figure V.13 avec la table V.9 associée.

O_k^-	R	$P(zo)$
0	0	0.01
0	1	0.01
1	0	obs
1	1	obs_{route}

TABLE V.9 – Table de probabilité conditionnelle suite au rajout de la contrainte d'être sur la route

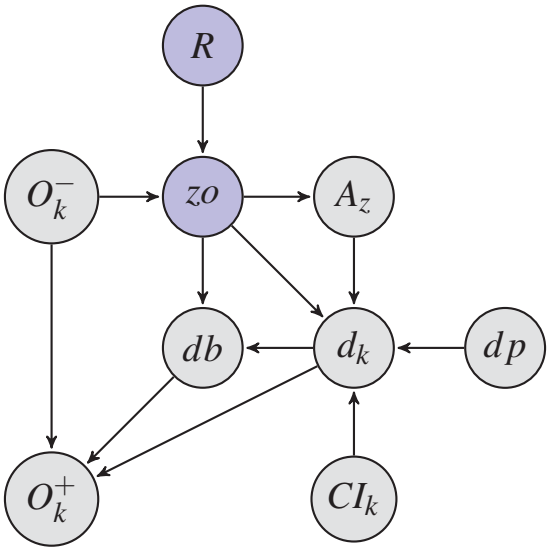


FIGURE V.13 – Réseau bayésien avec rajout de la contrainte d'être sur la route

Il est ainsi possible d'ajouter la prise en compte d'autres événements comme la présence d'amers non répertoriés dans la carte, la probabilité de présence d'un amer fourni par la carte (ce qui permettrait ainsi de mettre à jour cette dernière), le caractère opérationnel d'un capteur (défaillance ou non), etc.

V.3 Calcul des probabilités conditionnelles présentes dans le réseau bayésien

Dans un souci de clarté, le calcul de certains termes n'a pas été détaillé lors de la construction étape par étape du réseau. Ces termes sont maintenant présentés et expliqués ici plus en détail.

V.3.1 Observabilité

Dans V.2.3, la table de probabilité conditionnelle de z dépend de la variable obs qui représente la probabilité d'observer l'amer sachant que la position estimée est intègre. Comment calculer ce terme ? Le calcul de l'observabilité d'un amer doit prendre en compte deux phénomènes :

– l'observabilité due aux caractéristiques du capteur et de l'amer (l'amer est-il suffisamment grand pour être visible ? Le capteur est-il bien orienté ? À une distance suffisante ? ...) ;
– les occultations dues à la présence d'autres amers.

Le premier phénomène sera appelé l'observabilité intrinsèque du triplet, son calcul est effectué avec seulement les caractéristiques du triplet perceptif concerné. Le second sera appelé observabilité extrinsèque du triplet puisque son calcul dépend de l'environnement extérieur. Dans les approches existantes, l'observabilité extrinsèque est rarement prise en compte explicitement a priori (sauf dans quelques articles comme [88]) mais souvent a posteriori une fois que la détection a échoué en essayant de déterminer si cela est dû à une occultation ou non. Dans notre cas, il est possible de la prendre en compte explicitement grâce à notre connaissance partielle de l'environnement (connaissance de la position géoréférencée des autres amers) via la carte. Dans notre réseau bayésien, ces deux types d'observabilité sont utilisés pour calculer l'observabilité globale. Ce calcul est lié au type de capteur utilisé. Nous présentons deux cas de figure : un pour le GPS et un autre pour la caméra ou le télémètre laser.

Observabilité du GPS Le cas du GPS est particulier dans notre approche car il n'est pas associé à un amer, c'est un triplet dégénéré. Ceci n'est pas un problème pour nous car le système connaît uniquement le triplet perceptif et c'est le triplet perceptif qui lui fournit toutes les informations nécessaires. Le GPS n'est pas perturbé non plus par les autres amers de la même façon qu'un lidar ou une caméra. La qualité de la mesure fournie par le GPS va surtout être influencée par le nombre de satellites vus, la hauteur des murs et leur densité autour (phénomène de canyoning). Dans notre cas, nous avons seulement considéré que si nous étions en extérieur, le GPS donnait un résultat correct. Par conséquent, les environnements intérieurs sont considérés comme la seule source d'occultation du GPS ce qui implique que la probabilité d'observabilité du GPS soit égale à la probabilité que le véhicule soit dans un bâtiment. Le calcul de cette probabilité est facile et correspond au ratio entre l'intersection des zones intérieures et la zone d'incertitude. La figure V.14 illustre ce calcul, la probabilité d'observabilité sera le rapport entre la zone rouge et la zone hachurée.

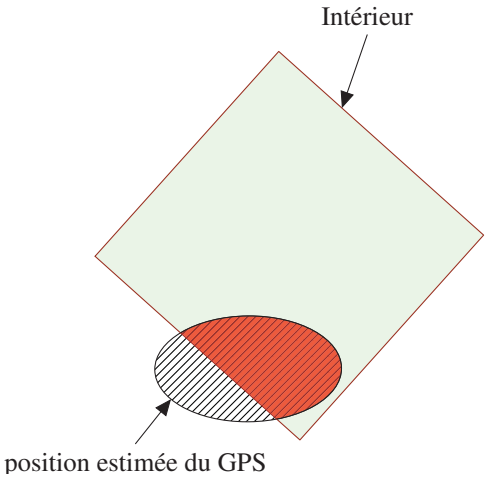


FIGURE V.14 – Observabilité du GPS

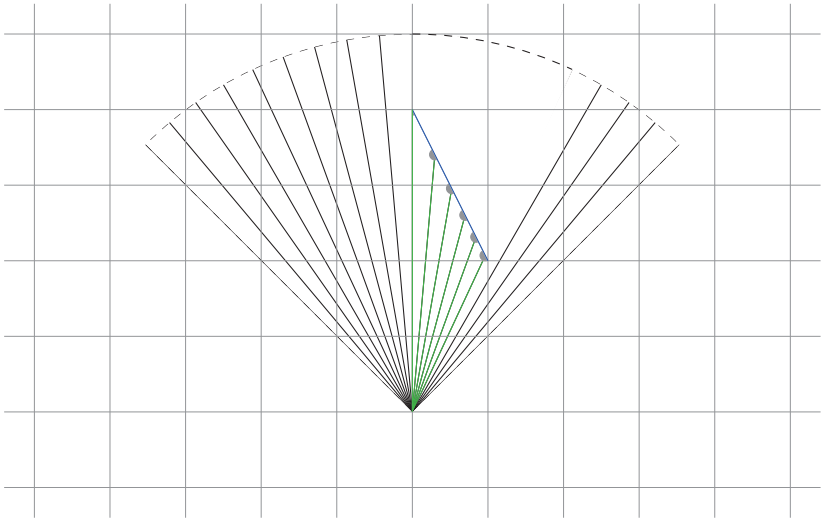


FIGURE V.15 – Estimation du nombre de points appartenant au mur dans les données lidar

Il serait simple de prendre en compte les occultations satellitaires par les murs par exemple mais ce travail n'a pas été réalisé dans le cadre de cette thèse.

Observabilité de la caméra ou du télémètre laser La méthode de calcul de l'observabilité est décrite ici pour un télémètre laser mais est facilement adaptable pour une caméra ou tout autre capteur extéroceptif. Pour des capteurs de ce type, l'observabilité dépend non seulement de la position du capteur et de l'amer recherché mais également des autres amers présents dans l'environnement. Pour mieux comprendre cette notion, considérons ici le triplet composé d'un mur, d'un télémètre laser et d'un détecteur de ligne. A partir de données fournies par le télémètre laser, sur quels critères pouvons-nous considérer qu'un mur est détectable avec le détecteur de ligne et les données fournies par un télémètre laser ? Nous considérons ici que seul le nombre de points appartenant au mur et renvoyés par le télémètre laser est important pour la détection. Plus le nombre de points renvoyés par le lidar appartiennent au mur, plus le détecteur aura de chances de le détecter correctement. La figure V.15 permet de comprendre le phénomène en jeu : seuls les points du mur atteints par les rayons sont pris en compte, la résolution angulaire du télémètre (qui détermine donc l'angle en les faisceaux), la distance entre le capteur et le mur ainsi que l'orientation et la longueur de ce dernier entrent directement en compte pour le nombre de points atteints.

Là-dessus, il faudra venir rajouter la présence d'obstacles qui peuvent cacher l'amer (par exemple, un autre mur présent devant celui recherché). Dans un premier temps, considérons seulement le triplet sélectionné sans aucun autre facteur extérieur. Le calcul de l'observabilité dans ces conditions revient à considérer le ratio entre la zone d'incertitude de la position estimée du capteur et la partie de cette zone d'où l'amer peut être visible.

Pour ceci, considérons l'exemple représenté sur la figure V.16. Notre véhicule est situé au point R. Dans un premier temps, nous supposerons que la position du véhicule est connue de manière précise sans incertitude. La position du capteur est confondue avec celle du véhicule. La zone de vision de ce capteur est représentée par un cône.

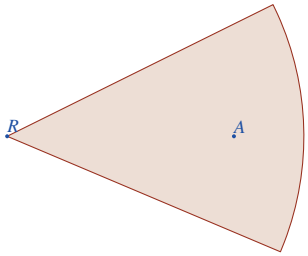


FIGURE V.16 – Exemple d'occultation : la position du robot est représenté par le point R et celle de l'amer par le point A

Si le capteur peut voir l'amer, nous pouvons considérer que l'amer "voit" le capteur par un raisonnement similaire au principe du retour inverse de la lumière. Il y a une symétrie entre les deux éléments. Nous allons donc considérer l'amer comme équipé d'un capteur lui permettant de visualiser le robot. Il faut donc définir la zone de vision de ce capteur, pour cela une symétrie centrale ayant pour centre de symétrie le milieu du segment $[AR]$ est réalisée. Le résultat est visible sur la figure V.17. Le cône hachuré correspond à la zone de visibilité de l'amer telle que définie précédemment.

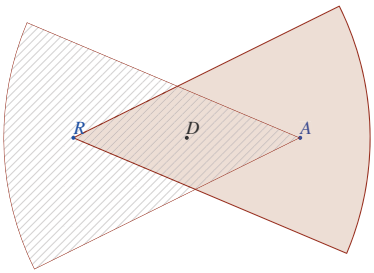


FIGURE V.17 – Zone de visibilité de l'amer obtenue par symétrie de centre D le milieu du segment $[RA]$

Maintenant, nous augmentons la difficulté du problème. La position du véhicule n'est plus considérée comme parfaitement connue mais avec une certaine incertitude modélisée par une matrice de covariance et représentée graphiquement par une ellipse (voir figure V.18). Certaines parties de cette zone d'incertitude sont à l'extérieur de la zone de vision de A, ceci implique par définition que le robot ne peut pas observer A s'il se trouve effectivement dans ces parties. La zone d'observabilité de A correspond donc à la zone qui est à la fois dans l'ellipse d'incertitude et dans la zone de vision de A. La probabilité d'observabilité de A se calcule donc immédiatement (voir équation V.1).

Dans la réalité, les amers qui ne sont pas réduits à un point sont visibles comme un ensemble de points et ceci quel que soit le capteur. Il n'est généralement pas nécessaire de voir l'objet en entier pour le détecter correctement, une partie de l'amer suffira. Par exemple, si nous essayons de détecter un mur avec un télémètre laser, une petite partie du mur est suffisante pour en déduire la distance à laquelle est le mur et comment il est orienté vis-à-vis du capteur ; pour détecter un panneau de sens interdit la partie centrale et une part suffisante du disque extérieur seront suffisants. C'est le détecteur qui nous

dira à partir de quand il arrivera à détecter l'amer. L'amer pourra être estimé comme étant une succession de points. Il faudra appliquer alors le même processus à chaque point.

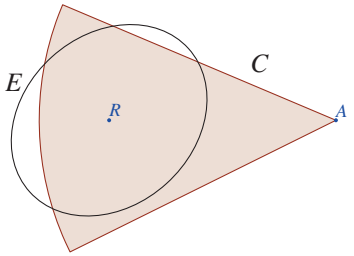


FIGURE V.18 – Calcul de l'observabilité intrinsèque : Prise en compte de l'incertitude. En bleu : les noms des positions, en noir : les noms des surfaces

$$obs(A) = \frac{\mathscr{A}(E \cap C)}{\mathscr{A}(E)} \tag{V.1}$$

Avec :
- $\mathscr{A}(E)$ l'aire de l'ellipse E ;
- $\mathscr{A}(E \cap C)$ l'aire de l'intersection de E et C.

La zone d'observabilité de l'amer en entier peut être calculée à partir des zones d'observabilité de chacun des points de l'amer en utilisant des règles particulières. Ces règles sont définies par notre détecteur et peuvent par exemple consister à ce qu'un mur est visible par un lidar si au moins trois points sont visibles. Il faut donc calculer la probabilité maximale de voir trois points en même temps (voir figure V.19 et l'équation V.2).

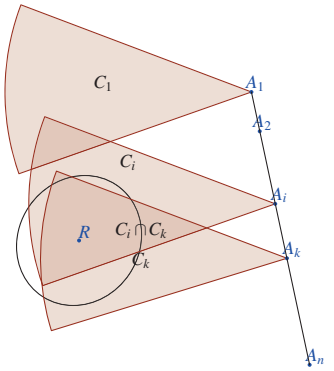


FIGURE V.19 – Calcul de l'observabilité intrinsèque pour un amer composé de plusieurs points

$$obs(A_i, A_l, A_k) = \frac{\mathscr{A}(E \cap C_i \cap C_l \cap C_k)}{\mathscr{A}(E)} \tag{V.2}$$

$$obs(Mur) = \max_{\substack{i,l,k \in 1,\ldots,n \\ i \neq l, i \neq k, l \neq k}} (obs(A_i, A_l, A_k)) \tag{V.3}$$

Maintenant que l'observabilité intrinsèque de l'amer a été calculée sans prendre en compte l'occultation possible par les autres amers, nous allons reprendre le même type de raisonnement mais en incluant cette fois l'occultation. Considérons une occultation par un mur tel que décrit sur la figure V.20.

Pour chaque point constituant l'amer, nous calculons la zone d'occultation de ce point par le tracé d'un cône ayant pour origine le point concerné et dont les demi-droites passent par les extrémités de l'amer vu sous l'angle du point. Cette étape est illustrée sur la figure V.20, la zone verte correspond à la zone d'où peut être le point courant (dont les coordonnées sont $(1,4)$), la zone rouge la zone d'où cela est impos-

sible à cause du mur occultant (situé entre les points $(-2.5,2)$ et $(0,2)$). Cette zone occultée est alors soustraite de la zone d'observabilité intrinsèque du point concerné. La zone résultante est donc la zone d'observabilité intrinsèque et extrinsèque. Le rapport de l'aire de cette zone et de l'aire de l'ellipse nous donne alors la probabilité d'observabilité du point. Le reste du calcul est alors effectué de manière similaire à celui de l'observabilité intrinsèque. Dans la réalité, il peut y avoir beaucoup de points appartenant à l'amer, il peut donc devenir rapidement chronophage de calculer toutes les combinaisons de points pour trouver la probabilité finale. Afin d'éviter ceci, nous effectuons un tirage aléatoire des différentes combinaisons possibles plutôt que de tout tester.

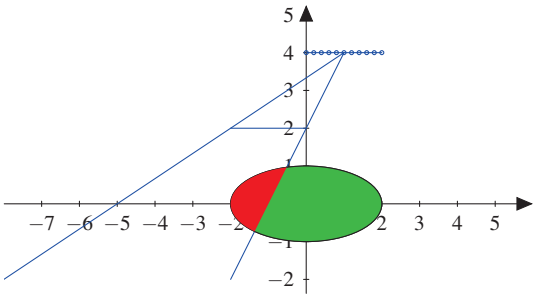


FIGURE V.20 – Calcul de la zone d'observabilité (en vert) pour chaque point appartenant à l'amer

Pour des raisons de rapidité également, l'observabilité extrinsèque (c'est-à-dire qu'un point est visible dès lors qu'il n'y a aucun obstacle entre le capteur et lui, sans prendre en compte l'orientation, la position du capteur, ...) est calculée en hors-ligne (figure V.21). Cette observabilité est stockée dans le Système d'Information Géographique (voir VII.2.6) utilisé et accessible via un système de requête.

V.3.2 Nombre de voisins

Le nombre de voisins dépend directement de la zone dans laquelle nous allons regarder, c'est-à-dire des données capteur qui vont effectivement être utilisées. Par exemple, dans une image, très souvent on ne regarde que la zone à l'intérieur d'une fenêtre rectangulaire appelée ROI (Region Of Interest). C'est ce que nous avons défini dans IV.4 comme étant la focalisation dans l'espace capteur. Cette zone dépend donc directement de notre méthode de détection (voir IV.4). Pour expliquer le calcul du nombre de voisins, nous considérerons la figure V.22 avec des amers-points. Leur position est supposée connue parfaitement (figure V.22a), ce qui n'est pas le cas de la position du véhicule.

L'incertitude de la position du véhicule est alors projetée sur chaque amer (figure V.22b). Deux amers sont alors considérés comme voisins spatialement si leurs ellipses d'incertitude sur la position s'intersectent (figure V.22c).

Dans le cas présenté (figure V.22), seule la position des amers est prise en compte pour déterminer s'ils sont voisins car c'est cette position que renvoie ici le détecteur.

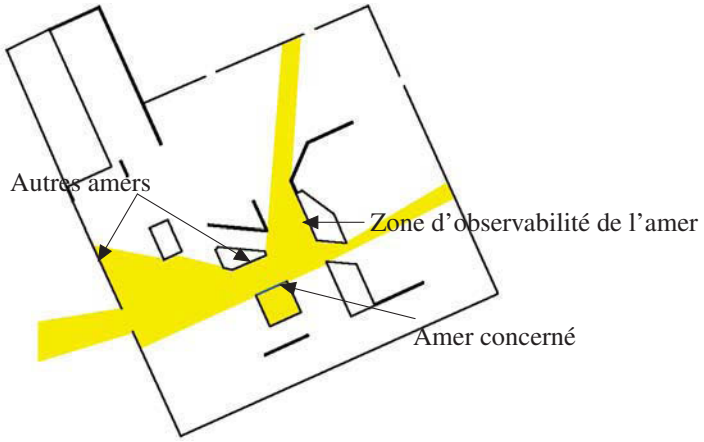
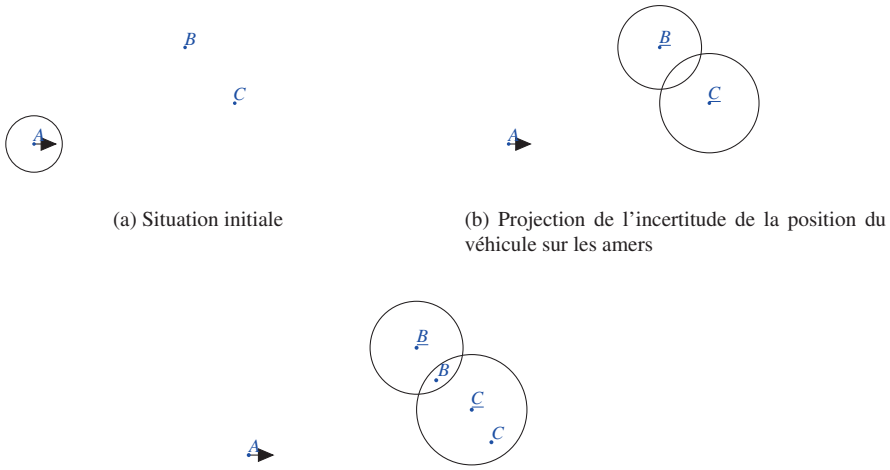


FIGURE V.21 – Calcul de l'observabilité d'un amer dans la base de données géoréférencée



(a) Situation initiale

(b) Projection de l'incertitude de la position du véhicule sur les amers

(c) Risque de confusion entre les deux voisins

FIGURE V.22 – Calcul du nombre de voisins possibles : Situation initiale

Ce n'est pas toujours le cas. Par exemple, pour des murs les paramètres renvoyés par le détecteur sont la distance et l'angle du mur depuis le point de vue du capteur. Dans ce cas, deux tests sont effectués : la proximité spatiale entre deux murs. Si effectivement leurs ellipses de focalisation s'intersectent, alors le même calcul est réalisé dans l'espace des paramètres. Si et seulement si ces deux tests sont positifs pour deux amers, alors les deux amers sont déclarés voisins.

V.3.3 Nombre de voisins issus de configurations similaires

Il peut arriver que des amers soient disposés de telle façon que des endroits différents se ressemblent, nous disons que les configurations sont similaires.

La situation présentée sur la figure V.8 montre deux configurations similaires. Ces configurations comportent deux éléments chacune (deux murs), la résolution de ce cas permet d'appréhender tout type de configuration comme le démontre le lemme suivant.

Lemme V.1 *Soient deux configurations, représentées par des suites notées* (U_n) *et* (V_n)*, composées d'amers. Alors savoir dire que deux configurations de taille* 2 *sont ou ne sont pas identiques permet de savoir si deux configurations de tailles quelconques supérieure ou égale à* 2 *sont identiques ou non.*

DÉMONSTRATION Nous pouvons montrer ce résultat grâce à un raisonnement par récurrence.

Initialisation : cas où $n = 2$ Le résultat est immédiat.

Passage de n **à** $n+1$

Soient $(U_n) = (u_1, u_2, \ldots, u_n)$ et $V_n = (v_1, v_2, \ldots, v_n)$, deux configurations similaires.

Alors (U_{n+1}) et (V_{n+1}) sont similaires si et seulement si les configurations (u_n, u_{n+1}) et (v_n, v_{n+1}) sont similaires, ce que nous pouvons savoir d'après les hypothèses car les configurations sont de dimension 2.

Il est donc possible de savoir si les configurations (U_{n+1}) et (V_{n+1}) sont similaires ou non. ■

Grâce à ce lemme, nous nous intéressons seulement aux configurations composées de deux amers. Nous procédons pour cela en deux temps : vérification spatiale et vérification dans l'espace des paramètres.

Première étape : vérification spatiale Considérons la figure V.23 qui reprend la situation de la figure V.8.

Le système sélectionne l'amer qu'il cherche à détecter, en l'occurence le mur 4, et va alors projeter son incertitude pour calculer la zone de focalisation (figure V.23a). Il va alors calculer le vecteur entre le centre de l'amer précédemment détecté (le mur 2) et le centre de la zone de focalisation (figure V.23b). Il va maintenant translater ce vecteur de manière à ce que son point de départ soit le centre des amers voisins de l'amer précédent et translater également la zone de focalisation (figure V.23c). Si des amers sont présents dans cette nouvelle zone alors il y a potentiellement une configuration similaire.

Seconde étape : vérification dans l'espace des paramètres Les triplets qui sont vérifiés sont ceux qui ont proviennent de l'étape de de vérification spatiale. Le processus est exactement le même mais cette fois dans l'espace des paramètres.

V.4 Modélisation par un réseau à chaque pas de temps

V.4.1 Introduction

Les réseaux bayésiens ne sont pas faits de base pour traiter des données séquentielles. Nous pensons spécialement ici aux données échelonnées dans le temps. Notre

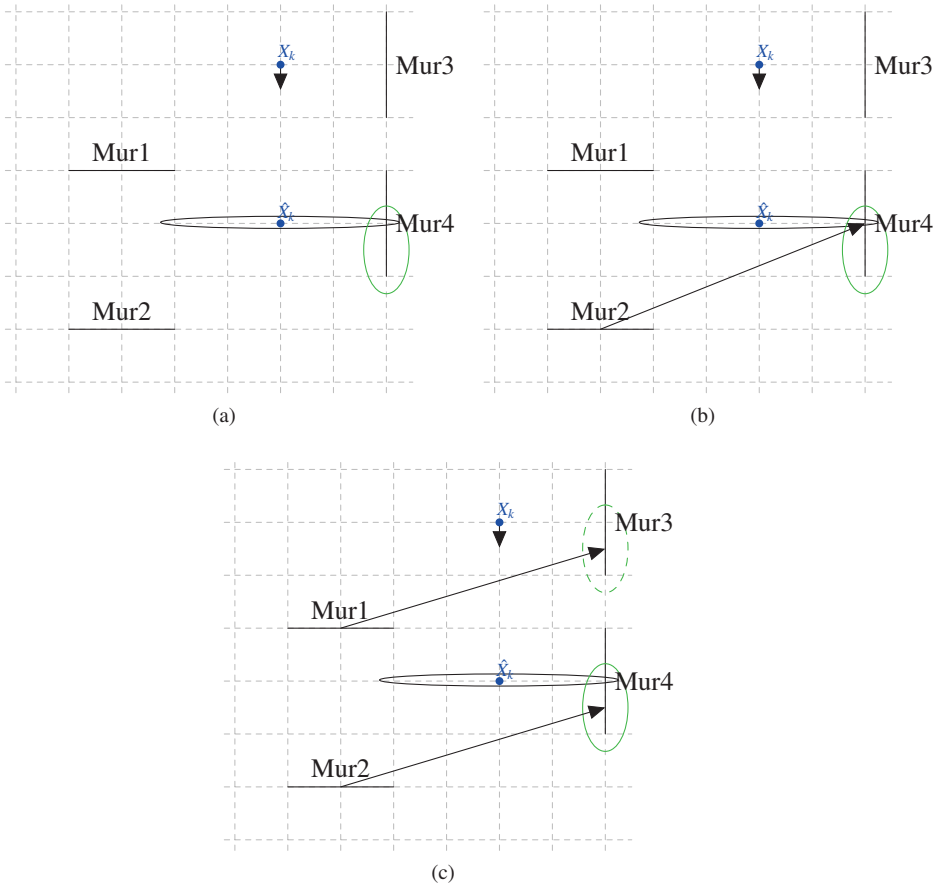


FIGURE V.23 – Exemple de configurations identiques : calcul de la compatibilité spatiale

système évolue dans le temps, ne serait-ce que parce que nous avons affaire à un robot mobile. Il faut donc prendre en compte cette information. Nous considérons ici le temps sous forme discrète. Le réseau bayésien créé dans les parties précédentes modélise un état figé à un instant k. A partir de là, comment le modéliser à l'instant $k+1$? Pour cela nous utilisons un réseau bayésien dynamique. Ce type de réseau est construit à partir d'un réseau bayésien classique en rajoutant un comportement temporel aux variables. A chaque instant k, une copie du réseau bayésien statique est réalisée pour modéliser cet instant précis, les tables de probabilités conditionnelles sont alors recalculées pour ce moment précis. En outre, au moins un nœud du réseau bayésien à l'instant $k+1$ a comme parent(s) au moins un nœud du réseau bayésien à l'instant k. Cela signifie également que nous faisons l'hypothèse que le système vérifie la propriété de Markov à savoir que l'état $k+1$ ne dépend que de l'état k.

V.4.2 Connexion entre les réseaux

Les deux réseaux modélisant les états k et $k+1$ sont reliés par les nœuds O_k^+ et O_{k+1}^- (figure V.24).

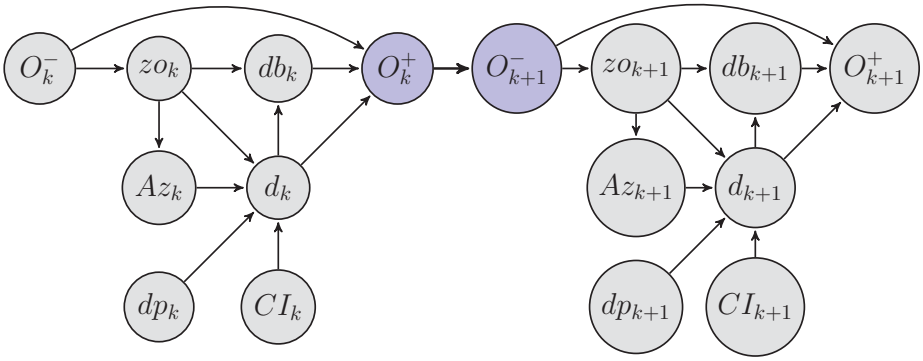


FIGURE V.24 – Réseau bayésien dynamique obtenu par chainage de réseaux bayésiens statiques

O_k^+	$P(O_{k+1}^-)$
0	0
1	1

TABLE V.10 – Table de probabilité conditionnelle permettant de modéliser le passage d'un instant k à un instant $k+1$

Ces nœuds ne modélisent pas obligatoirement le même état. En effet entre deux instants, le véhicule bouge et son déplacement est calculé en prenant en compte l'odométrie et l'angle au volant via un modèle d'évolution mis en place à travers un filtre de Kalman. Nous considérons ici que l'intégrité de la localisation estimée est maintenue grâce à la bonne caractérisation des bruits sur les capteurs proprioceptifs du véhicule. La table de probabilité conditionnelle est alors très simplifiée (Tableau V.10).

Le passage entre deux instants est ainsi simplifié puisqu'il suffit de reporter dans le réseau bayésien de l'étape $k+1$ la probabilité de O_k^+ comme valeur a priori de $P(O_{k+1}^-)$, cette simplification ne serait plus vraie si le robot dérapait par exemple (il serait possible de prendre éventuellement ce phénomène en compte, dans ce cas le tableau V.10 ferait apparaître les probabilités correspondantes). Ceci est notamment très utile puisque pour chaque état, nous pouvons utiliser un réseau bayésien spécifique et non associer l'ensemble des réseaux bayésiens connectés depuis l'instant initial. Il est aisé de comprendre que ceci est d'une grande aide au niveau calculatoire puisqu'à chaque instant, seul un réseau bayésien d'une dizaine de nœuds suffit et ne risque donc pas d'être trop coûteux pour l'application.

V.4.3 Inférence

Réseau à l'instant k

Un des grands avantages d'un réseau bayésien est de pouvoir inférer de nouveaux faits, à partir d'observations. Prenons l'exemple suivant avec la neige tombée durant la nuit. Les événements suivants sont définis :

Veille la neige était présente la veille ;

Nuit la neige est tombée durant la nuit ;

Sol la neige est présente sur le sol.

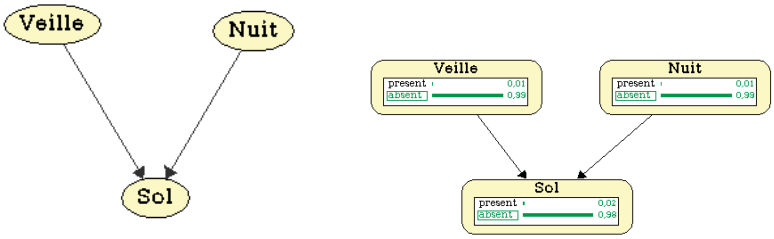
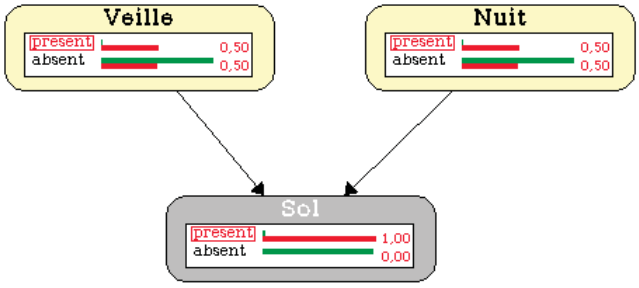
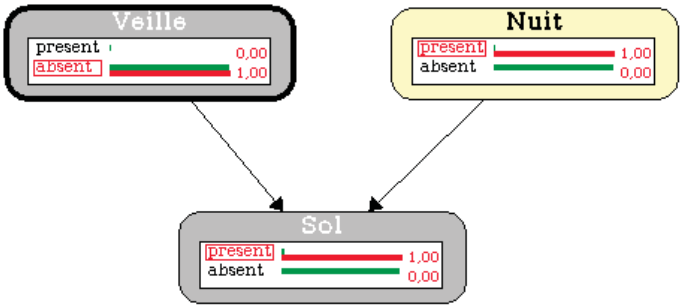


FIGURE V.25 – Réseau bayésien modélisant la tombée de la neige (figure réalisée à l'aide du logiciel Elvira [34])



(a) Rajout de l'observation : de la neige est présente sur le sol



(b) Rajout de l'observation : de la neige est présente sur le sol et il n'a pas neigé hier

FIGURE V.26 – Propagation d'inférence à travers le réseau

Le logiciel Elvira [34] nous permet de visualiser facilement les valeurs des probabilités des différents événements (figure V.26) et leur évolution lorsqu'une évidence est rajoutée, enclenchant ainsi un processus d'inférence. Si nous observons qu'il y a de la neige sur le sol, alors les probabilités qu'il ait neigé la veille ou cette nuit augmentent (figure V.26a). Si en plus l'observation « il n'a pas neigé hier » est rajoutée alors la

probabilité qu'il ait neigé cette nuit est de 100% (figure V.26b).

Le calcul d'inférence est réalisé suivant différents algorithmes. On cite par exemple l'algorithme JLO pour *Jensen, Lauritzen, Olesen* [62] qui est un algorithme d'inférence exacte, c'est cet algorithme qui est utilisé dans notre cas (voir annexe B). Il existe également des algorithmes de calcul d'inférence approchée, cela peut être utile notamment pour de très gros réseaux,+- ce qui n'est pas notre cas. Ces algorithmes effectuent d'abord un fort travail de regroupement des différents nœuds à l'aide de la théorie des graphes, permettant de diminuer le temps de calcul.

Réseau dynamique regroupant les instants de 0 à k

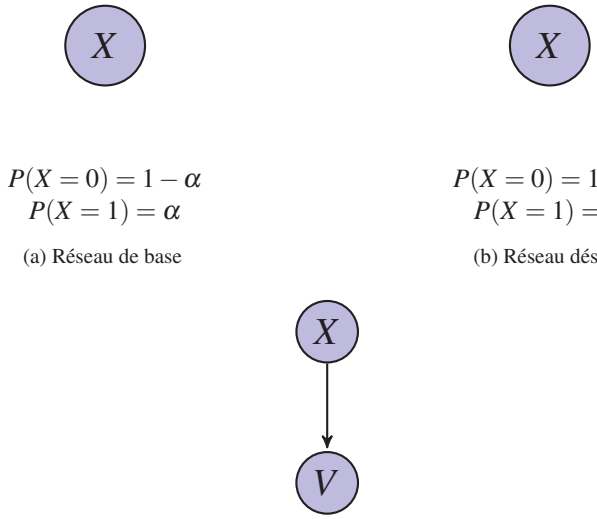
L'inférence dans un réseau bayésien dynamique suit exactement le même processus que pour un réseau bayésien statique. Cela oblige à considérer le réseau dans son ensemble depuis l'instant initial jusqu'à l'instant présent. Cela peut être très fastidieux et pose toujours un souci dans les applications utilisant des réseaux bayésiens dynamiques [135]. Dans notre cas, il n'est pas envisageable de stocker la structure complète du réseau au fur et à mesure qu'elle s'accroît. Chaque réseau bayésien correspondant à un instant donné est stocké (la durée de stockage peut être fixée par un autre processus) indépendamment des autres. S'il est toujours possible d'initialiser la probabilité a priori O^-_{k+1} avec la valeur de $P(O^+_k)$, il n'en va pas de même lorsque l'on désire faire de l'inférence.

En effet, plaçons-nous dans la situation où une évidence a été observée à l'instant $k+1$, cette information a été inférée dans le réseau à l'instant $k+1$ mais nous désirons savoir ce que cela implique sur le réseau à l'instant k. Dans notre cas, la jonction entre les deux réseaux est très simple puisqu'elle est réalisée à travers le fait que $P(O^-_{k+1}) = P(O^+_k)$, aussi il suffirait d'obliger $P(O^+_k)$ à avoir la même valeur que $P(O^-_{k+1}|evidence)$ pour réaliser l'inférence de manière correcte. Nous pourrions affirmer que nous voulons forcer la probabilité d'un nœud à une certaine valeur, en d'autres termes d'avoir une probabilité comme évidence. Cette notion n'est pas nouvelle dans les réseaux bayésiens. Dans [114], Pearl définit pour cela la notion d'évidence virtuelle qui est vue comme une généralisation de l'évidence classique des réseaux bayésiens.

Un exemple d'application classique donné dans [19] peut être le suivant : soit une variable X pouvant prendre trois valeurs parmi $\{1,2,3\}$, généralement une évidence sur ce nœud sera du type $X = 2$ mais nous pouvons avoir une observation partielle qui nous indique seulement que $X \neq 1$, un réseau bayésien classique ne pourra pas prendre cette information en compte. Dans notre cas, nous désirons pouvoir fixer la probabilité d'un nœud X à une valeur différente de 0 ou 1. Pour cela, nous définissons un nouveau nœud V virtuel configuré de telle sorte qu'une évidence sur ce nœud force la probabilité du nœud X à la valeur voulue.

Pour comprendre la solution adoptée, prenons la figure V.27. Tous les événements définis dans ce réseau bayésien sont des événements binaires.

Sur cette dernière, le nœud tel qu'il est au départ est représenté, c'est-à-dire que la probabilité que l'événement X soit vrai est de α. Dans l'application, le but est d'avoir la probabilité de cet événement fixée à la valeur β. Pour cela, un nœud virtuel modélisant un événement virtuel V est rajouté de manière à contraindre cette probabilité (figure V.27c). Ce nœud va être fixé comme étant une inférence, d'où le nom d'évidence virtuelle adopté dans [114]. Afin d'avoir le comportement souhaité, il faut donc définir les probabilités conditionnelles de l'événement V. Ces probabilités conditionnelles vont être calculées à l'aide du théorème qui suit.



$P(X=0) = 1-\alpha$
$P(X=1) = \alpha$

(a) Réseau de base

$P(X=0) = 1-\beta$
$P(X=1) = \beta$

(b) Réseau désiré

$P(X=0|V=1) = 1-\beta$
$P(X=1|V=1) = \beta$

(c) Moyen mis en œuvre : rajout d'un nœud virtuel V

FIGURE V.27 – Illustration du procédé utilisé pour fixer la probabilité d'un événement à une valeur prédéfinie dans le réseau bayésien dynamique

Théorème 3
Soit l'événement binaire X. Soit $\alpha \in]0,1[$ la valeur de la probabilité a priori que l'événement X soit égal à 1. Soit $\beta \in [0,1]$ la valeur désirée de la probabilité de l'événement X. Soit γ tel que $0 < \gamma < \min(\frac{\alpha}{\beta}, \frac{1-\alpha}{1-\beta})$. Soit l'événement binaire V. Alors :

X	$P(V=1)$
0	$\gamma \times \frac{1-\beta}{1-\alpha}$
1	$\gamma \times \frac{\beta}{\alpha}$

$\Longrightarrow P(X=1|V=1) = \beta$

DÉMONSTRATION Soit $\gamma \in [0, \min(\frac{\alpha}{\beta}, \frac{1-\alpha}{1-\beta}]$. Soit l'événement V dont les probabilités conditionnelles sont définies dans le théorème. Soit α et β tels que définis dans le théorème. On démontre de manière triviale que ces probabilités sont contenues entre 0 et 1 grâce aux conditions sur γ.

D'après la formule des probabilités totales :

$$\begin{aligned} P(V=1) &= P(V=1|X=0)P(X=0) + P(V=1|X=1)P(X=1) \\ &= \gamma\frac{1-\beta}{1-\alpha}\alpha + \gamma\frac{\beta}{\alpha}\alpha \qquad \text{par définition de } V \text{ et de } \alpha \\ &= \gamma \end{aligned}$$

D'après le théorème de Bayes, nous avons :

$$\begin{aligned} P(X=1|V=1) &= \frac{P(V=1|X=1)P(X=1)}{P(V=1)} \\ &= \frac{\gamma\beta}{\alpha}\frac{P(X=1)}{P(V=1)} \qquad \text{par définition de } V \\ &= \frac{\gamma\beta}{\alpha}\frac{\alpha}{P(V=1)} \qquad \text{par définition de } \alpha \\ &= \frac{\gamma\beta}{\alpha}\frac{\alpha}{\gamma} \qquad \text{par utilisation du fait que } P(V=1)=\gamma \end{aligned}$$

D'où $P(X=1|V=1)=\beta$ ■

Grâce à ce théorème, nous savons qu'il suffit de choisir une valeur γ dans l'intervalle défini dans le théorème pour en déduire les probabilités de l'événement V conditionnellement à l'événement X d'après les formules données dans le tableau pour avoir le comportement désiré.

Le processus peut ainsi être répété de proche en proche comme illustré sur la figure V.28.

V.5 Conclusion

Nous avons présenté ici pourquoi la modélisation des interactions entre les différents événements était cruciale. Les différents points qui ont été abordés sont les suivants :

- construction de notre réseau bayésien pas à pas ;
- levée des hypothèses idéales permettant ainsi la modélisation d'un monde réel ;
- description des réseaux bayésiens dynamiques ;
- description de la méthode d'évidence virtuelle utilisée.

Les techniques de calcul à l'intérieur des réseaux bayésiens n'ont pas été décrites dans ce manuscrit. Ces techniques sont cependant loin d'être triviales et peuvent être variées notamment pour des réseaux bayésiens de grande taille, le lecteur intéressé pourra se reporter à [27, 114, 135] pour plus d'informations. Dans l'annexe B, l'algorithme utilisé dans notre cas est décrit.

L'utilité d'un tel réseau bayésien a déjà été démontrée grâce au calcul de la probabilité de bonne détection utilisée dans le critère de sélection. Le réseau bayésien est également utilisé pour le calcul de la confiance et la prise en compte du résultat de la détection. Ceci est utilisé lors de la phase de mise à jour de la position et de la confiance détaillée dans le chapitre suivant.

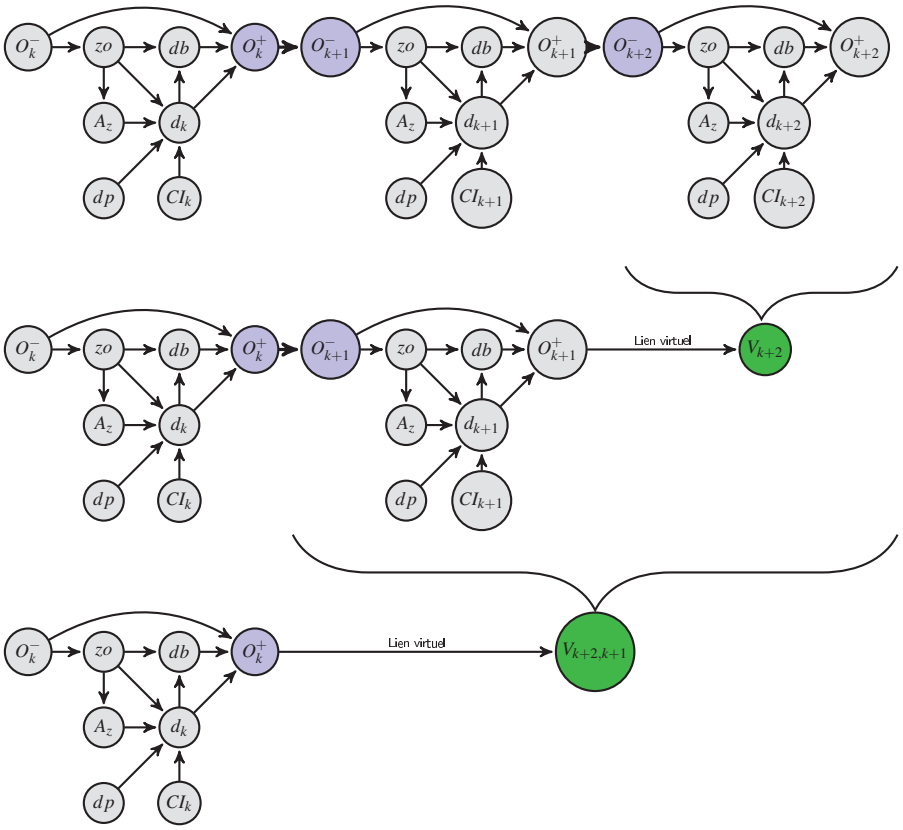


FIGURE V.28 – Processus d'inférence de proche en proche développé

CHAPITRE VI

Processus de mise à jour

Sommaire

VI.1 Introduction

Dans les chapitres précédents, les étapes de sélection et de détection de l'approche proposée ont été présentées. Le processus doit maintenant mettre à jour l'état du système à partir des informations fournies par le détecteur. Cette mise à jour concerne deux données : la position du robot et la confiance dans cette position. Deux informations sont fournies par le détecteur : d'une part le fait que la détection ait réussi ou échoué, et d'autre part si la détection est réussie, le détecteur fournit une seconde information qui correspond aux paramètres de l'amer détecté (comme la distance et l'orientation d'un mur par rapport au capteur par exemple) et de la précision de cette mesure. Dans les deux cas, le détecteur fournit au moins une information qu'il est nécessaire de prendre en compte dans le processus de mise à jour de la position et de la confiance qui est décrit dans la suite de ce chapitre.

VI.2 Détection réussie

Le premier cas concerne la réussite de la détection. Cela signifie que le détecteur a renvoyé un résultat positif. Deux mises à jour s'imposent alors : l'état modélisant la position du robot d'une part et la confiance dans cette estimation d'autre part. Si la détection a réussi, cela signifie que le détecteur a renvoyé des données compatibles avec le triplet recherché (mais il n'est pas certain que le détecteur ait détecté l'amer recherché).

VI.2.1 Mise à jour de la position

Ces données doivent être fusionnées avec l'état prédit à l'instant $k+1$ connaissant toutes les mesures jusqu'à l'instant k, noté $X_{k+1|k}$, déjà connu du véhicule, pour fournir une nouvelle estimation de la position du véhicule et de la matrice de covariance associée. Cette nouvelle estimation de l'état $k+1$ connaissant la mesure $k+1$ est notée $X_{k+1|k+1}$. La position est mise à jour ici grâce au filtre de Kalman décrit dans l'annexe A.

VI.2.2 Mise à jour de la confiance

Indépendamment de la mise à jour de la position, la réussite du détecteur est déjà une information en soi qui nous renseigne sur l'état du robot. Initialement, le réseau bayésien a été construit en estimant a priori, à partir de la carte, les triplets qui pouvaient être confondus avec le triplet recherché caractérisant ainsi le nombre de voisins compatibles. A l'issue de la phase de détection, il est possible de refaire cette estimation avec beaucoup plus de précision à partir des informations fournies par le détecteur.

Ensuite le résultat positif de la détection est utilisé grâce au principe d'inférence du réseau bayésien. Le nœud correspondant à l'événement « une détection a été réussie » est le nœud d_k. Ce nœud devient donc une évidence et le réseau bayésien va utiliser cette information pour mettre à jour les autres probabilités du réseau (figure VI.1).

Ainsi, la confiance sur l'état du véhicule est mise à jour. Il est important de noter qu'ici il n'y a pas d'information sur l'adéquation entre le triplet recherché et le résultat du détecteur. Il est tout à fait possible que cette association soit fausse. Cette information est prise en compte dans le réseau bayésien.

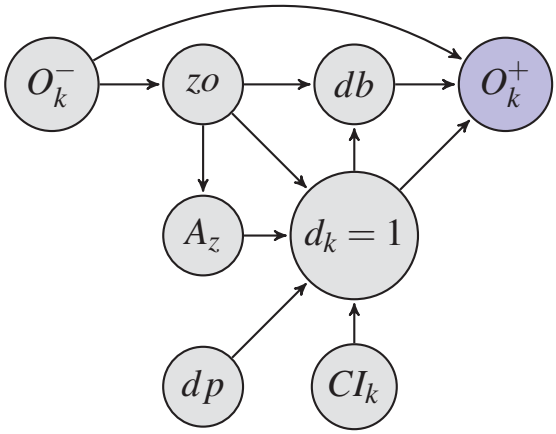


FIGURE VI.1 – Inférence dans le réseau bayésien : l'évidence $d_k = 1$ est utilisée pour mettre à jour le réseau et en particulier la probabilité $P(O_k^+)$ correspondant à la confiance en la nouvelle position estimée du robot lorsqu'on a mis à jour son état avec la détection réalisée.

VI.3 Détection échouée

Malgré le processus de sélection et de focalisation qui a été conçu pour maximiser les chances de détecter un triplet, la détection de ce dernier peut échouer. Dans ce cas puisque le détecteur n'a pas renvoyé de données, il est inutile de mettre à jour à l 'aide du filtre de Kalman l'estimation de la pose du véhicule. En revanche, la confiance doit être mise à jour de manière similaire au cas où la détection réussit.

VI.3.1 Mise à jour de la valeur de la confiance

Afin de recalculer les probabilités des différents événements, une évidence ($d_k = 0$) est rajoutée dans le réseau bayésien de manière similaire au cas où la détection réussit.

VI.3.2 Identification de la source d'erreur

Plusieurs raisons peuvent entraîner un échec de la détection. Il est important d'identifier la source d'erreur la plus probable afin que l'algorithme adopte un comportement adéquat. Par exemple si la raison expliquant l'échec de la détection provient du fait qu'un piéton occultait l'amer réellement recherché, la position estimée ne doit pas être remise en cause. Si en revanche, le détecteur a échoué parce que le robot n'est pas à l'endroit estimé alors cela doit être pris en compte rapidement car aucune autre tâche ne pourra être exécutée correctement par le robot. Les sources d'erreur peuvent être définies à partir de deux événements du réseau bayésien : zo (l'amer est présent dans la zone de recherche et il est visible) et O_k^- (La position estimée est intègre). Nous classons les sources d'erreur en trois parties :

a) $\overline{zo} \cap O_k^-$: l'amer recherché ne peut pas être vu car il est occulté par un autre amer présent dans la carte ;

b) zo : le détecteur n'a pas trouvé l'amer alors que celui est bien présent dans la zone de recherche et qu'il n'y a pas d'autre amer de la carte qui puisse l'occulter. Cette

cause regroupe les cas suivants :
- L'amer est occulté parce qu'un amer non répertorié est occultant ;
- le détecteur ou le capteur sont défaillants ;
- un obstacle provisoire était présent (comme un piéton par exemple), ;
- l'amer est répertorié dans la carte mais n'est pas présent en réalité ;
- etc.

c) $\overline{O_k^-}$: Cet événement est bien évidemment le plus important. Le détecteur a échoué car le robot n'est pas à la position estimée.

Tous ces événements et leurs probabilités associées sont donnés sachant que la détection a échoué, c'est-à-dire sachant que $P(d_k) = 0$. A chaque instant et à partir des informations disponibles il va falloir estimer la source d'erreur la plus probable parmi celles listées précédemment. Ainsi, les probabilités liées à ces différentes sources vont être comparées. Ces probabilités sont calculées grâce à la mise à jour du réseau bayésien en prenant en compte le fait que la détection a échoué (inférence de l'événement $dk = 0$). Deux cas de figures se présentent :
- $P(\overline{O_k^-}) < P(\overline{zo} \cap O_k^-)$ ou $P(\overline{O_k^-}) < P(zo)$
- $P(\overline{O_k^-}) > P(\overline{zo} \cap O_k^-)$ et $P(\overline{O_k^-}) > P(zo)$

Dans le premier cas, le fait que le robot ne soit pas au bon endroit n'est pas l'hypothèse la plus probable. L'erreur provient plus certainement d'un défaut du détecteur ou d'une occultation. Dans cette situation, le processus continue normalement les étapes de sélection, de détection du triplet et de mise à jour de l'état. Dans le second cas, le robot n'a pas pu détecter l'amer recherché et c'est très probablement lié au fait que sa position estimée est fausse. Dans ce cas, il faut être capable de revenir à une position estimée intègre (position réelle incluse dans l'incertitude de la position estimée) et il est nécessaire de développer un système de correction des erreurs.

VI.3.3 Correction des erreurs

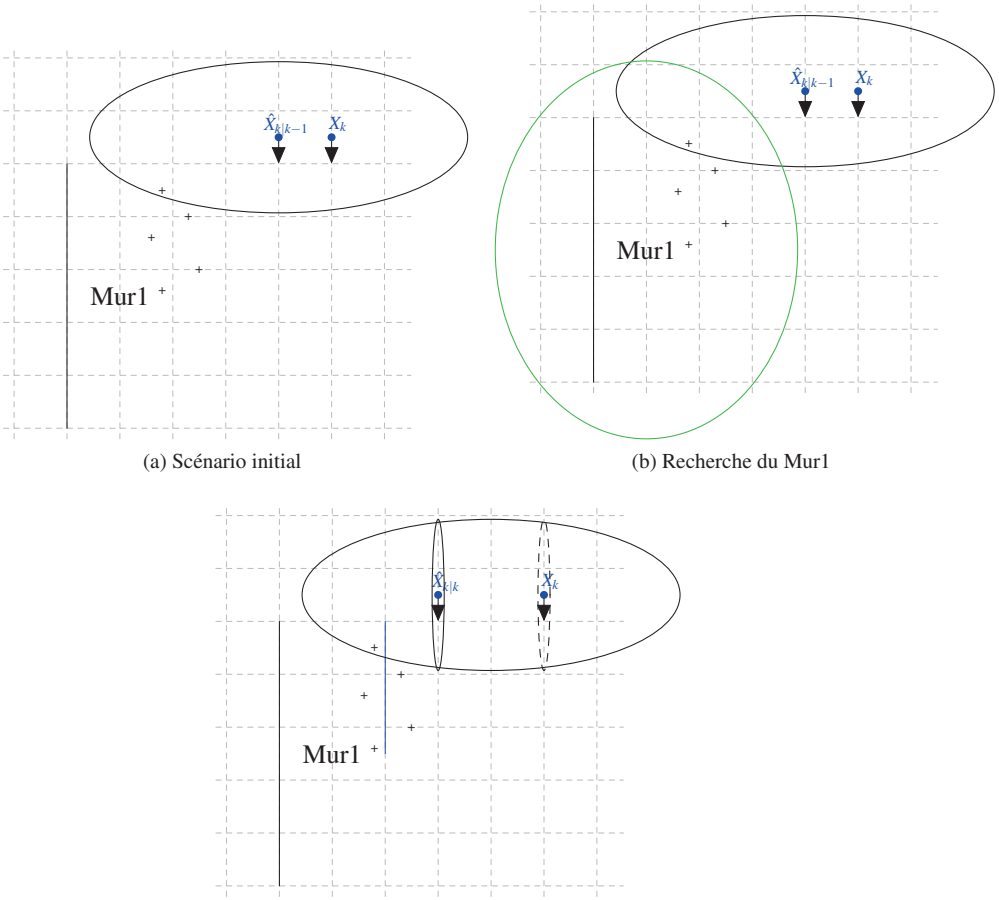
La position estimée par le processus peut être fausse en raison d'une précédente mauvaise association. Il faut donc éliminer ou réparer cette mauvaise association. Cette mauvaise association peut avoir deux causes : soit le détecteur a renvoyé un résultat ne correspondant à aucun amer de la carte et a donc été perturbé par la présence de bruit (amer non référencé, objet mobile, etc), soit il a renvoyé un résultat correspondant à un autre amer de la carte. Ces deux cas de figure sont étudiés ci-dessous.

Correction d'une mauvaise association due à la présence de bruit

Dans le cas d'une mauvaise association due à la présence de bruit dans les données capteur comme la présence de buissons détectés à la place d'un mur, il est important d'éliminer cette mauvaise association. A partir de là, nous pourrons repropager l'ancienne hypothèse.

Pour bien comprendre ce principe de remise en cause, considérons les figures VI.2 et VI.3. Dans un premier temps la position estimée du véhicule est intègre (figure VI.2a). Cela signifie que l'ellipse d'incertitude centrée sur la position estimée contient la position réelle. Le seul amer disponible est le Mur 1 qui est donc sélectionné. La zone de focalisation pour rechercher le Mur 1 est définie à partir de la projection dans l'espace capteur de l'incertitude de la position estimée (figure VI.2b).

Malheureusement la présence de bruit (causé par la présence de buissons) dans l'image capteur fait que le mur n'est pas correctement détecté. Tout se passe comme si



(a) Scénario initial

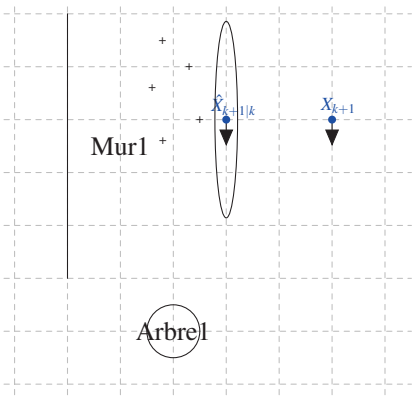
(b) Recherche du Mur1

(c) Mise à jour erronée suite à une erreur de détection. Un faux mur (en bleu) est détecté.

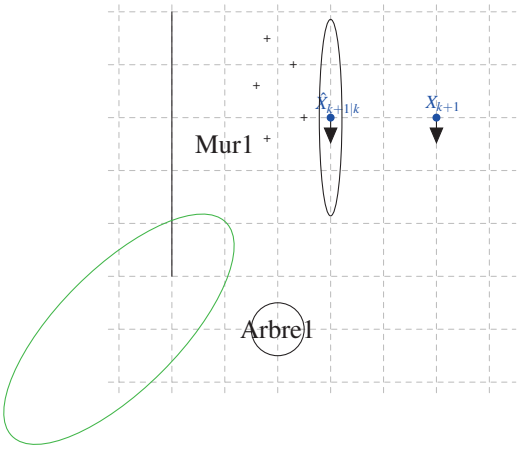
FIGURE VI.2 – Mauvaise association due à la présence de bruit : mise en situation. Un buisson non repertorié est symbolisé par des croix.

un mur parallèle était détecté en lieu et place de celui recherché. Un filtre de Kalman permet de remettre à jour la position du robot en fonction de cette détection. Ainsi une nouvelle estimation de la pose est disponible (figure VI.2c). Cette position n'est plus intègre : la position réelle n'est pas dans l'ellipse d'incertitude centrée sur la position estimée. A ce moment-là, le robot n'a pas de moyen de détecter ce problème puisqu'aucun autre triplet n'est disponible.

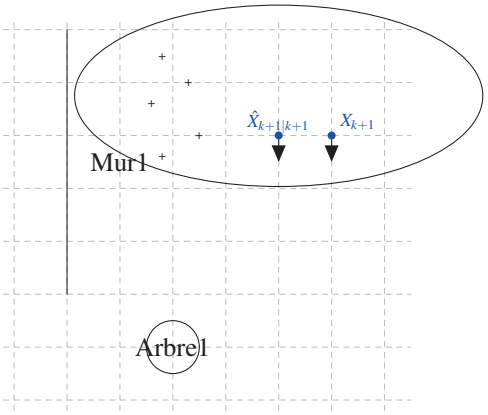
Le robot avance dans son environnement. Durant ce déplacement, les capteurs proprioceptifs du véhicule sont utilisés afin de réaliser une étape de prédiction du filtre de Kalman. La position du véhicule continue d'être estimée mais avec une incertitude de plus en plus grande. Un autre triplet devient alors potentiellement détectable (figure VI.3a). Le robot va essayer alors de détecter ce nouveau triplet (Arbre1). La position estimée n'étant pas intègre, la zone de focalisation n'englobe pas le lieu du triplet à détecter (figure VI.3b). Dans cette situation la détection échoue. Le système cherche alors à déterminer la cause de cet échec. Dans cet exemple, la cause de l'échec est at-



(a) Déplacement du robot et découverte d'un nouveau triplet disponible



(b) Échec de détection de ce nouveau triplet



(c) Remise en cause par élimination de l'association précédente, retour à une position intègre

FIGURE VI.3 – Mauvaise association due à la présence de bruit : correction de l'erreur.

tribuée à une mauvaise association précédente. Cette association est alors supprimée. Le modèle d'évolution permet d'avoir une nouvelle position estimée à l'instant courant. Une calibration grossière des capteurs proprioceptifs, pris en compte dans ce modèle d'évolution, permet de conserver une estimation intègre (figure VI.3c).

Correction d'une mauvaise association due à une ambiguïté

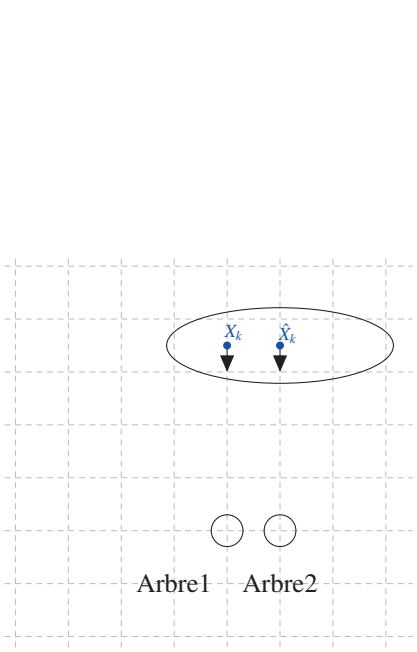
Dans la zone de recherche définie par le processus de focalisation, d'autres amers cartographiées (voisins) compatibles avec celui recherché peuvent être présents. Dans ce cas-là, la mauvaise association est plus probablement liée à une hypothèse dont nous avions déjà connaissance au moment de la détection plutôt qu'à un défaut du détecteur ou à la présence d'un amer non répertorié.

Pour résoudre cette situation, nous pourrions agir comme dans le cas d'une mau-

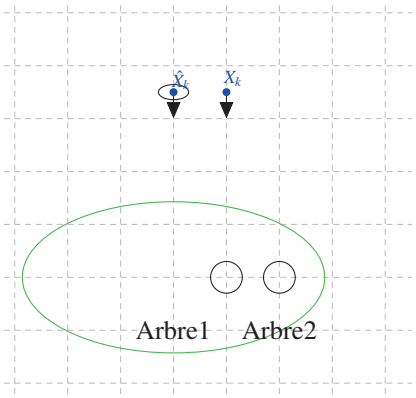
vaise association due à la présence de bruit c'est-à-dire éliminer cette association puis repropager l'état jusqu'à l'instant courant et ainsi conserver l'intégrité. Cependant cela présente deux inconvénients :

– risque de situation de blocage ;
– retour à une position intègre mais avec une grande incertitude alors que la détection d'un amer cartographié est disponible.

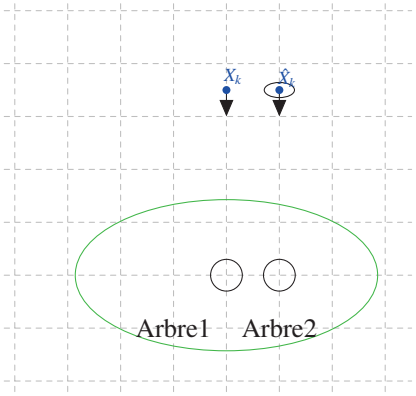
Dans le premier cas, le problème est le suivant : Il peut arriver qu'en désirant détecter un amer A, on détecte l'amer B et inversement. Par exemple sur la figure VI.4a, deux amers Arbre1 et Arbre2 sont potentiellement visibles. Quand le système essaie de détecter Arbre1 (figure VI.4b), la zone de focalisation englobe à la fois Arbre1 et Arbre2 ; Arbre2 est détecté et la détection est associée à Arbre1 (puisque c'était lui qui était recherché), cette association donne alors lieu à une position estimée non-intègre (la position réelle n'est pas à l'intérieur de l'ellipse autour de la position estimée). De manière similaire, si le système essaie de détecter Arbre2 (figure VI.4c), il détecte en réalité Arbre1.



(a) État initial



(b) Recherche de Arbre1 : détection de Arbre2 et mise à jour



(c) Recherche de Arbre2 : détection de Arbre1 et mise à jour

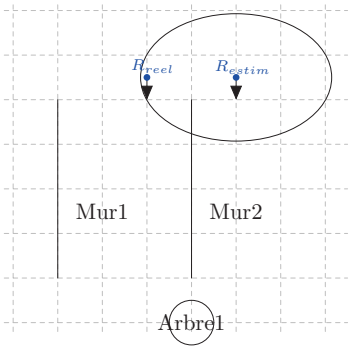
FIGURE VI.4 – Situation de blocage : que nous cherchions à détecter un amer ou l'autre la position estimée n'est pas intègre

Lors de l'essai de détection de l'amer recherché, le système sait que ce dernier a

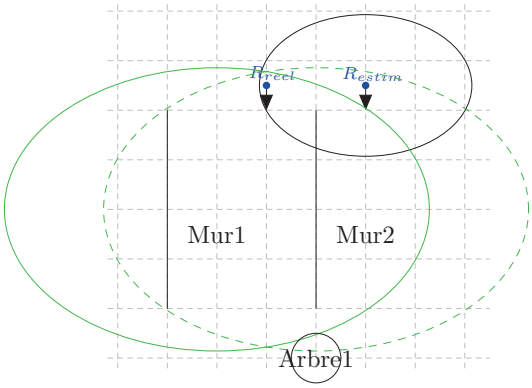
un ou des voisin(s) potentiel(s) et que donc leur détection va faire chuter la confiance, cette chute de confiance va permettre de savoir que l'association était fausse et qu'il faut donc la remettre en cause. Cependant, le système risque d'osciller indéfiniment et rien ne permet d'affirmer que cette oscillation va cesser. Nous sommes dans une situation d'inter-blocage.

Dans le second cas, sans la nécessité de faire une nouvelle détection, une donnée est disponible et il serait dommage de ne pas l'exploiter. Cette détection étant déjà réalisée, l'exploiter permettrait de réduire des temps de calcul (pas besoin de relancer le processus de sélection/focalisation/détection). De plus, les données issues du capteur correspondant à l'instant de cette détection ne sont plus disponibles sauf si elles étaient toutes enregistrées ce qui s'avère quasiment impossible (espace mémoire rédhibitoire). Du fait que le processus de détection a déjà été réalisé, les associations possibles entre l'amer recherché et les détections obtenues dans la zone parcourue sont connues.

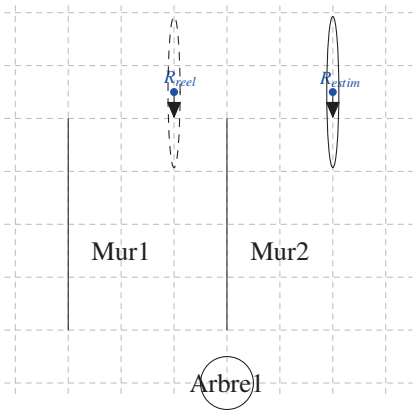
Ainsi, une nouvelle association entre l'amer recherché et une autre détection réalisée va être envisagée ce qui permettra de reprogager l'état jusqu'à l'instant présent. Pour cela, le système bénéficie de la liste des voisins, c'est-à-dire de la liste des amers compatibles avec le résultat du détecteur. Si cette liste est vide, alors la mauvaise association est purement et simplement supprimée, nous retombons dans la situation du cas précédent. Si la liste n'est pas vide, l'observation est associée avec le premier voisin. Le premier voisin est défini comme étant le triplet pour lequel la distance de Mahalanobis entre les caractéristiques renvoyées par le détecteur et les caractéristiques attendues de l'amer avec la covariance projetée dans l'espace des paramètres est minimale. Le processus se déroule ensuite normalement. Pour bien comprendre ce qui se passe considérons la figure VI.5.



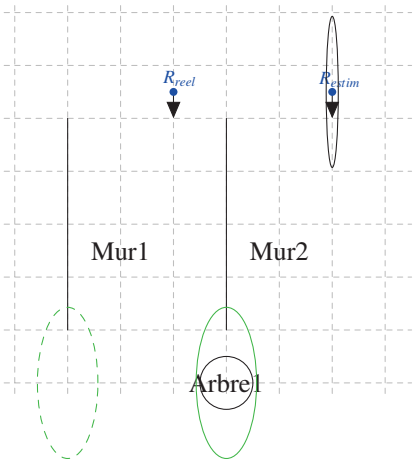
(a) Position initiale



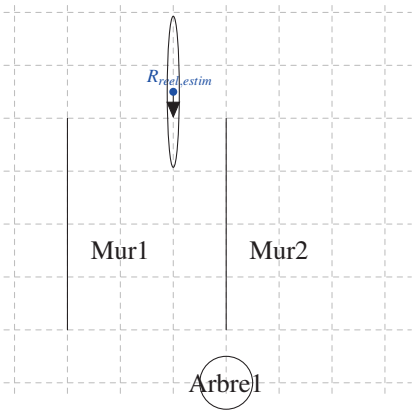
(b) Recherche du Mur2, détection du Mur1.



(c) Association de l'observation du Mur1 avec le Mur2



(d) Recherche de Arbre1, échec de la détection



(e) Réassociation de la première observation avec Mur1, la position redevient intègre

FIGURE VI.5 – Correction d'une mauvaise association : cas d'une ambiguïté. La zone de focalisation réelle est représentée par une ellipse verte en pointillé, la zone de focalisation estimée par le système est représentée par une ellipse verte en trait plein.

Dans un premier temps, la position estimée est intègre (figure VI.5a). Deux amers sont visibles, Mur1 et Mur2. Le système sélectionne alors le Mur2, la zone de focalisation (ellipse verte) est grande et englobe également le Mur1 (figure VI.5b). C'est effectivement le Mur1 qui est détecté et qui est associé avec le Mur2 initialement recherché. Cette mauvaise association due à une ambiguïté implique une position non intègre après mise à jour (figure VI.5c). Le système sélectionne alors Arbre1 qui ne se trouve pas dans la zone de recherche réelle (ellipse verte) et qui n'est pas détecté (figure VI.5d). La probabilité liée à la confiance sur l'estimation a donc fortement chuté et le processus de remise en cause est alors enclenché. Nous supposons ici pour l'exemple, que l'événement « la position n'est pas intègre » est décelé comme étant le plus probable suite à l'inférence de l'événement « la détection de l'amer Arbre1 a échoué ». L'association précédente ayant été réalisée avec un triplet possédant des voisins, le système sait qu'il a peut être associé l'observation avec le mauvais amer. Cette association est donc remise en cause et une nouvelle association est réalisée cette fois avec l'amer Mur1 (figure VI.5e). Le processus peut ensuite suivre son cours normal avec sélection de Arbre1 par exemple.

Détermination de l'association à remettre en cause

Nous avons vu dans les deux sections précédentes quelles pouvaient être les sources d'erreurs entraînant une mauvaise estimation de la pose du véhicule et comment y remédier. Nous avons également décrit une méthode pour remédier au problème d'une fausse association. Le problème qui se pose est de déterminer quelle est cette mauvaise association. Une première idée peut être de toujours réparer ou éliminer la dernière mauvaise détection, de repropager l'état et de continuer le processus. C'est le système développé et utilisé dans [142]. De proche en proche, nous finirons par arriver à la mauvaise association et à revenir ainsi dans le droit chemin. Cette méthode sera souvent très efficace puisque généralement la mauvaise association sera souvent la dernière (c'est par exemple le cas dans l'exemple de la figure VI.5 précédemment présenté).

Cependant il peut arriver que l'erreur se propage suite à un problème de corrélation des données. Si nous observons régulièrement un amer car il est le seul disponible, les détections successives seront corrélées et si la première fois une mauvaise association a été réalisée, elle risque de se propager. Ceci est illustré par la figure VI.6. A l'étape k, la position estimée est intègre mais assez imprécise ce qui implique une focalisation peu discriminante lors de la détection d'un triplet. Le système essaie de détecter le Mur 2 mais détecte le Mur 1 et réalise ainsi une mauvaise association. Le véhicule se met ensuite en mouvement, durant tout son mouvement le seul triplet disponible est alors le Mur 1 qu'il croit être le Mur 2 et détecte donc comme tel. Arrivé au moment où un autre triplet est disponible, l'arbre 1, le système échoue dans la détection et choisit de se remettre en cause. S'il remet en cause uniquement la dernière association, c'est-à-dire en l'occurence qu'il la supprime, alors la position estimée sera toujours non-intègre. Il faut en réalité remettre en cause toutes les associations réalisées entre celle effectuée à l'instant k et l'instant $k+4$.

Il faut donc plutôt éliminer toutes les observations précédentes si elles sont corrélées jusqu'à revenir à la dernière qui ne l'est pas ou pour laquelle il y avait une ambiguïté.

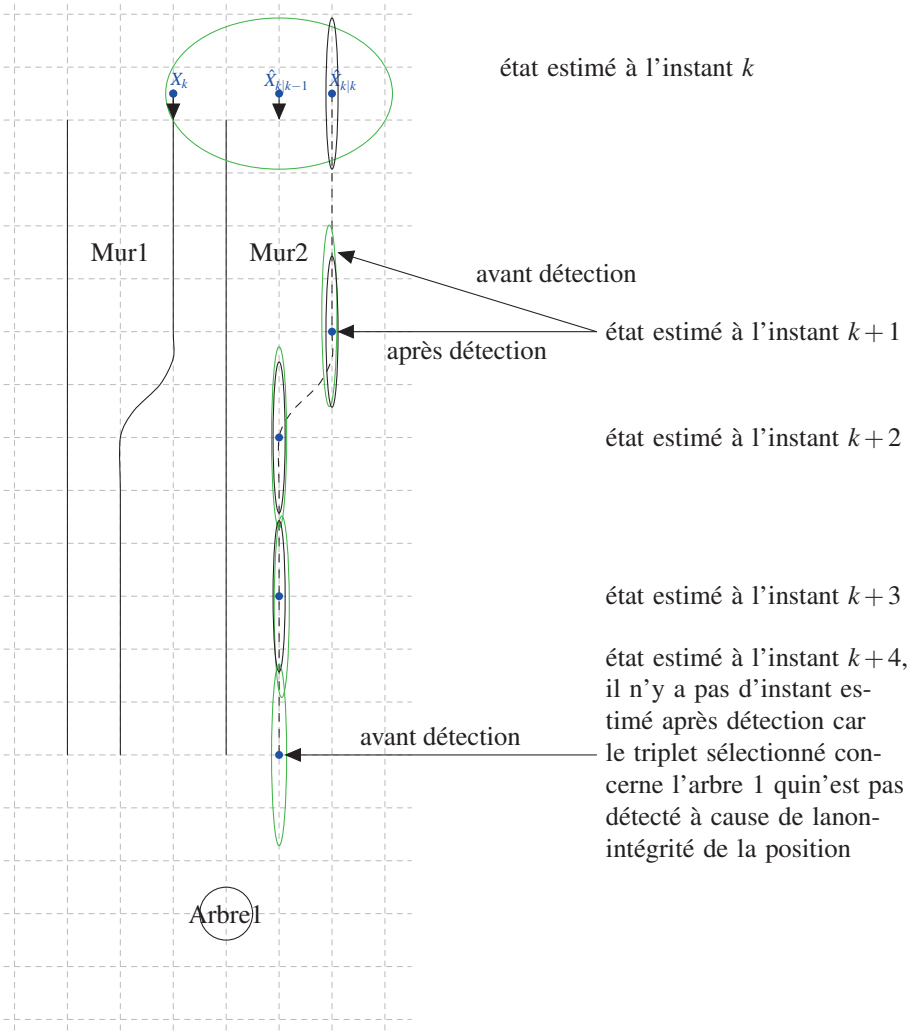


FIGURE VI.6 – Illustration de l'influence des données corrélées sur une détection

VI.4 Bilan

Nous avons vu tout le processus de mise à jour de la position et de la confiance avec prise en compte des possibles erreurs d'association et les différents moyens d'y remédier. Tout ceci peut être résumé par l'algorithme 1.

Il nous reste maintenant à voir comment l'information de corrélation, qui est utilisée à la fois dans le réseau bayésien et pour déterminer quelle association doit être remise en cause, est mise à jour pour chaque triplet.

VI.5 Mise à jour de la corrélation entre les données

Dans les chapitres précédents, nous avons vu comment la distance parcourue depuis la dernière tentative (réussie ou non) de détection d'un triplet était utilisée pour calculer

Algorithm 1. Mise à jour et remise en cause éventuelle

Ensure: Backtracking
  Calcul de $P(\overline{O_k^-})$, $P(zo)$, $P(\alpha)$
  if $P(\overline{O_k^-}) < P(zo)$ **or** $P(\overline{O_k^-}) < P(\alpha)$ **then**
    Pas de remise en cause des observations {L'échec de la détection n'est probablement pas dû à une non-intégrité}
    return
  else
    if Triplet précédent avait des voisins **then**
      Re-association de l'observation avec le premier voisin
    else
      Remonter à la dernière détection d'un triplet non-corrélé ou avec des voisins
    end if
  end if
  Mise à jour des probabilités

le degré de corrélation de ce dernier avec les dernières détections réalisées. Cela signifie qu'à chaque fois qu'un triplet est détecté, il est conservé en mémoire de manière à pouvoir calculer ultérieurement son degré de corrélation. Cependant dans certaines situations, il faudra supprimer de la mémoire certaines de ces détections. Par exemple, si le processus de détection avait échoué pour détecter un amer car notre position n'était pas intègre, le triplet composé notamment de cet amer n'a jamais pu nous fournir d'information donc celle-ci ne peut être corrélée. Si maintenant, nous avons à nouveau une position intègre, nous pouvons donc essayer de le détecter et bénéficier de toute l'information qu'il peut apporter. Cinq cas de figure différents peuvent se présenter et sont détaillés ici ainsi que la solution proposée. Chaque cas est présenté à travers un exemple illustré à l'aide de tableaux comprenant les champs suivants :

Instant : ce champ donne le numéro de l'étape concernée. La première étape est notée k, les suivantes $k+1$, $k+2$, etc.

Observation : ce champ comporte le résultat de la détection. Si la détection a réussi, l'observation résultante est notée obs_k si nous sommes à l'étape k, obs_{k+1} si nous sommes à l'étape $k+1$, etc. Si la détection a échoué, cela sera symbolisé par le symbole $\emptyset$.

Arbre de propagation : dans ce champ sont donnés, sous la forme d'arbre, les triplets que le système a essayé de détecter au cours du temps et qui ont contribué à l'estimation de l'état actuel (position et confiance) ainsi que leurs voisins potentiels. Les triplets que le système a essayé de détecter sont situés sur la gauche de l'arbre. Pour un triplet, les voisins éventuels sont situés sur la même ligne et ont le même parent.

Associations : seule la branche gauche de l'arbre du champ « Arbre de propagation » est présente ici. Pour tous ces triplets, l'observation qui leur a été associée est indiquée, si une détection a échouée cela est symbolisé par $\emptyset$.

Liste des triplets stockés : A chaque étape, ce champ donne la liste des triplets pour lesquels une détection a été tentée ou une association réalisée et qui ont contribué à l'état actuel du robot (confiance et position). Suite aux différentes remises en cause effectuées, des éléments de cette liste peuvent être supprimés ou échangés avec d'autres triplets. En sus, pour chaque triplet est stocké en mémoire la dis-

tance parcourue par le robot depuis l'essai de détection de ce triplet. Cela permet au système de pouvoir calculer la corrélation de l'information éventuellement fournie par ce triplet avec les précédentes détections s'il est resélectionné.

VI.5.1 Cas numéro 1

Cette situation est la plus facile. Le système sélectionne des triplets à détecter. Les détections successives réussissent et les triplets correspondants sont gardés en mémoire.

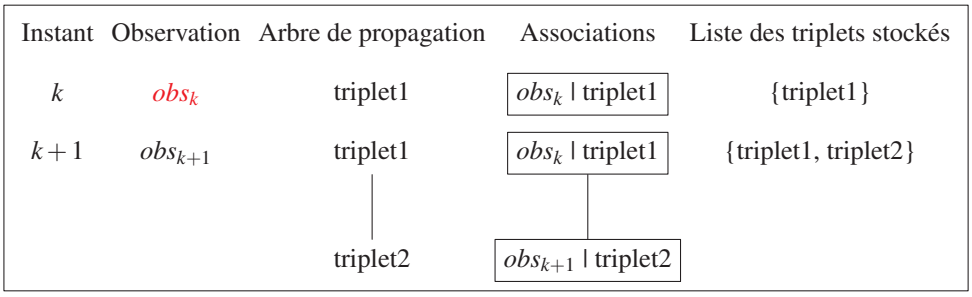


FIGURE VI.7 – Stockage des différents triplets sélectionnés en vue du calcul de la corrélation des données entre elles : Cas 1

Ce cas est présenté sur la figure VI.7. Les détections aux instants k et $k+1$ sont réussies et sont alors associées aux triplets 1 et 2. Ces derniers sont stockés en mémoire de manière à pouvoir connaître ultérieurement la distance parcourue depuis leurs dernières détections.

VI.5.2 Cas numéro 2

Ce cas est un peu plus complexe car une détection a échoué mais il n'y a pas eu de remise en cause des étapes précédentes. Le système conserve donc en mémoire la tentative ratée de détection de ce triplet.

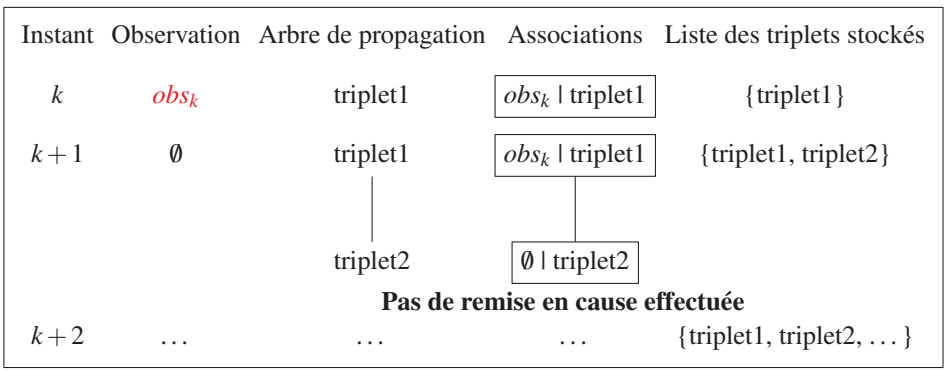


FIGURE VI.8 – Stockage des différents triplets sélectionnés en vue du calcul de la corrélation des données entre elles : Cas 2

Ce processus est illustré par un exemple décrit sur la figure VI.8. A l'étape k, le système essaie de détecter le triplet 1 et associe alors le résultat de la détection obs_k au

triplet 1. Le triplet 1 est donc stocké. A l'étape $k+1$, le triplet 2 est recherché. La détection échoue, il n'y a donc pas d'association effectuée. Ce triplet est stocké en mémoire comme ayant fait l'objet d'une tentative de détection. Comme la détection a échoué, la question se pose de savoir s'il faut remettre en cause l'association précédente. Dans cet exemple, il n'y a pas de remise en cause. La liste reste donc inchangée. Le processus se poursuit ensuite normalement.

VI.5.3 Cas numéro 3

Nous nous trouvons ici dans une situation presque similaire à la précédente. Une détection a échoué mais cette fois le système de remise à cause a déterminé que cet échec était probablement dû à une erreur d'association lors des précédentes observations. L'observation précédente est alors remise en cause. Cette remise en cause se traduit ici par une suppression de l'association précédente. Le système ne conserve donc pas en mémoire le fait que la détection a échoué car cela provient certainement du triplet parent et non du triplet pour lequel il y a eu un échec. En revanche, le triplet qui avait été mal associé est conservé en mémoire.

Instant	Observation	Arbre de propagation	Associations	Liste des triplets stockés
k	obs_k	triplet1	obs_k | triplet1	{triplet1}
$k+1$	$\emptyset$	triplet1	obs_k | triplet1	{triplet1, triplet2}
		triplet2	$\emptyset$ | triplet2	
Remise en cause effectuée				
$k+1$	...	...	$\emptyset$ | triplet1	{triplet1}

FIGURE VI.9 – Stockage des différents triplets sélectionnés en vue du calcul de la corrélation des données entre elles : Cas 3

La figure VI.9 illustre ce cas précis. A l'étape k, une observation obs_k est disponible et associée au triplet 1. A l'étape $k+2$, le triplet 2 est sélectionné et une tentative de détection a lieu. La détection est un échec et le système détermine que l'association précédente doit être remise en cause. La remise en cause consiste ici à supprimer l'association entre l'observation obs_k et le triplet 1. Il est important de savoir que la détection de ce triplet 1 risque de conduire à une mauvaise association. Il est donc nécessaire de le conserver en mémoire. En revanche, la détection du triplet 2 ayant échoué à cause de l'association précédente avec le triplet 1, celle-ci étant effacée il n'y a pas de raison de conserver le triplet 2.

VI.5.4 Cas numéro 4

Le cas est similaire au précédent à la différence que la remise en cause de l'observation précédente se traduit ici par une réassociation de l'observation car le triplet associé possédait un voisin potentiel. Il ne faut donc pas conserver en mémoire le triplet initialement associé car ce n'est peut-être pas lui qui était observé. L'observer ultérieurement après la réassociation pourrait donc devenir possible notamment car l'espace de

focalisation serait plus petit et éliminerait le risque de le confondre de nouveau avec son voisin.

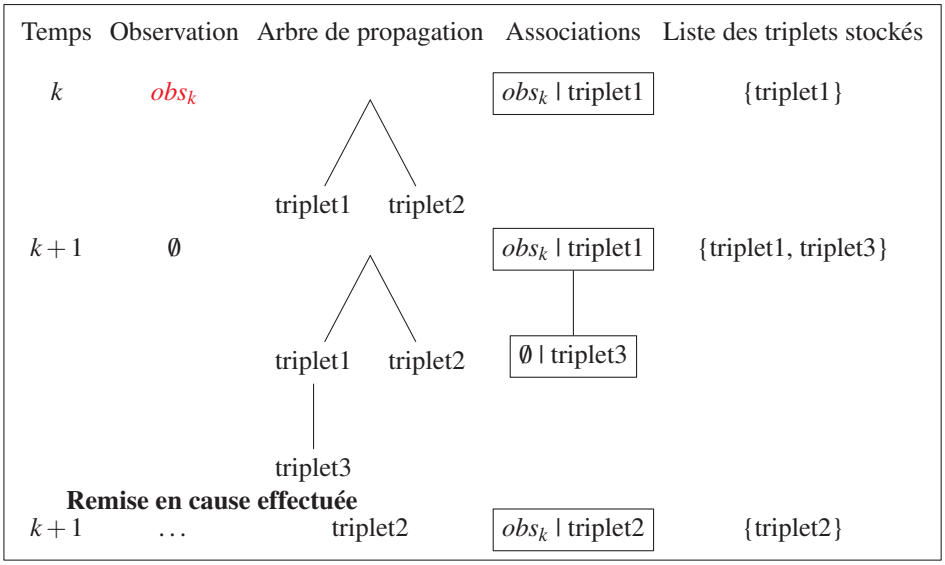


FIGURE VI.10 – Stockage des différents triplets sélectionnés en vue du calcul de la corrélation des données entre elles : Cas 4

La figure VI.10 illustre ce cas précis. A l'étape k, une observation obs_k est disponible et associée au triplet 1, sachant qu'un autre triplet est potentiellement compatible, le triplet 2. Le triplet 1 est stocké en mémoire. A l'étape $k+1$, le triplet 3 est sélectionné et une tentative de détection a lieu. La détection est un échec et le système détermine que l'association précédente doit être remise en cause. La remise en cause consiste à associer l'observation obs_k au triplet 2 plutôt qu'au triplet 1. Les triplets 1 et 3 sont alors enlevés de la liste de stockage pour être remplacés par le triplet 2. Le processus se poursuit ensuite normalement.

VI.5.5 Cas numéro 5

Ici le processus se déroule au départ comme dans le cas 4. Cependant, après la remise en cause du cas 4, un nouvel essai de détection échoue et déclenche un nouveau processus de remise en cause. Cela signifie que l'association qui est remise en cause est déjà une association qui avait été réaffectée. Deux cas peuvent se présenter :

1. Il existe encore un(des) voisin(s) compatible(s) ;
2. il n'existe plus de voisin compatible.

Dans le premier cas, l'observation est réassociée au triplet voisin suivant et dans la liste des triplets stockés le triplet désormais associé remplace celui qui l'était précédemment. Dans le second cas, l'association est supprimée. Il faut donc retenir que le triplet initial qui avait été sélectionné n'a pas pu être associé correctement. Ce cas est illustré sur la figure VI.11.

Les étapes k et $k+1$ se déroulent de la même manière que dans le cas 4. A l'étape $k+2$, la détection du triplet sélectionné, à savoir le triplet 3, échoue. Le système estime qu'il faut remettre en cause l'association précédente, à savoir l'association de obs_k avec

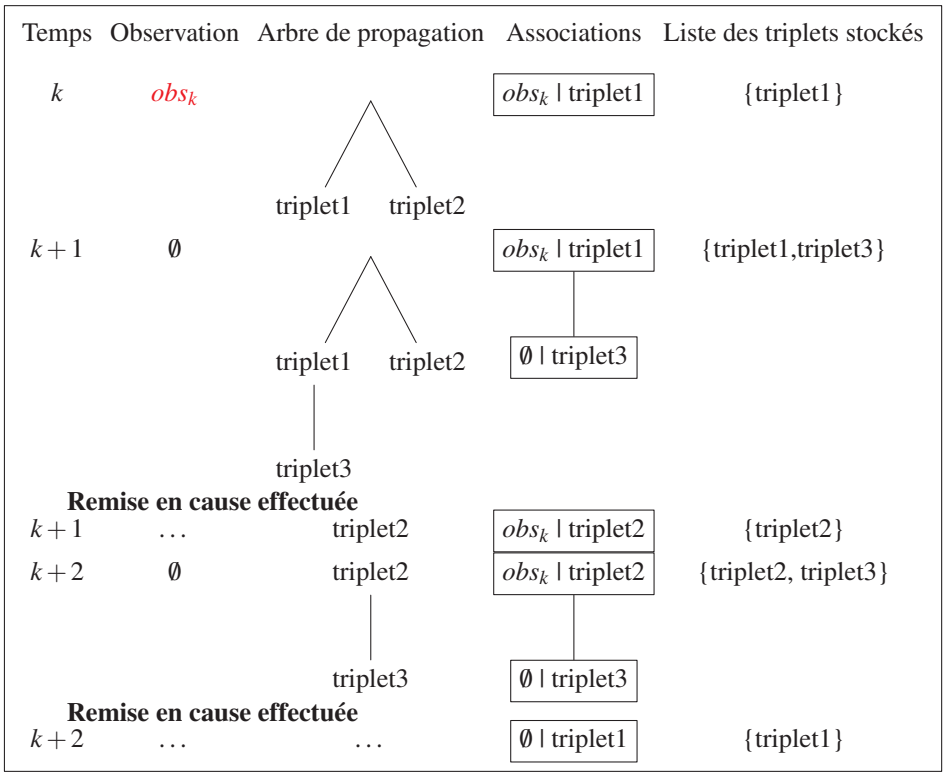


FIGURE VI.11 – Stockage des différents triplets sélectionnés en vue du calcul de la corrélation des données entre elles : Cas 5

le triplet 2. Étant donné qu'il n'y a pas d'autre triplet voisin, cette association est donc éliminée. C'est alors le premier triplet qui avait été sélectionné qui est intégré dans la liste de stockage pour le calcul de la corrélation, c'est-à-dire le triplet 1. En effet, il résulte de tout ceci que l'observation obs_k qui a été créé à l'occasion de l'essai de détection du triplet 1 n'a pu être associée correctement à aucun triplet, elle doit donc être éliminée. Le triplet 1 n'ayant pu être détecté doit être stocké en mémoire comme tel afin que plus tard le système sache qu'il a essayé de le détecter mais que cela a échoué.

VI.6 Conclusion

Le processus de mise à jour après une détection ou une non détection a été présenté ici. Cela permet d'avoir à tout instant une estimation sur la localisation et l'intégrité. Si cela est nécessaire un processus de remise en cause d'associations passées a été développé afin de garantir l'intégrité du système à chaque instant. Par ailleurs, ce processus permet de résoudre le problème bien connu du robot kidnappé. En effet, si nous supposons au départ de l'algorithme global que l'incertitude du robot est égale à la taille de la carte et donc qu'au départ notre position estimée est intègre. Par sa capacité à réassocier les différentes observations, notre système est capable, à condition d'avoir un nombre suffisant d'amers disponibles, d'atteindre ses objectifs de précision et d'intégrité à partir d'une incertitude initiale très grande. Des exemples de résultats

de cette application du robot kidnappé seront présentés dans la partie suivante. Il est important de noter que notre système compte tenu de ses caractéristiques est identique tout au long du parcours du robot. En effet, contrairement à la majorité des approches existantes, aucune étape spécifique d'initialisation n'est nécessaire.

CHAPITRE VII

Expérimentations et résultats

Sommaire

VII.1 Introduction

Dans les chapitres précédents, le principe de l'approche Top-Down développée, comprenant la sélection du triplet perceptif le plus pertinent et le processus de remise en cause mis en place, a été décrit. Cette méthode de localisation a été implémentée et testée en situations simulées et réelles. Pour cela, deux grandes parties sont présentées dans ce chapitre :

- l'environnement : quel est le matériel utilisé, les choix techniques qui ont été faits, etc ;
- les résultats : afin de valider notre algorithme, différentes mises en situation sont effectuées afin de voir les caractéristiques et le comportement global de l'application. L'algorithme est testé en situation statique et dynamique. Des résultats montrant comment l'algorithme développé peut être utilisé pour de la conduite autonome seul ou en convoi sont également présentés.

Enfin une dernière partie sera consacrée à un bilan de notre approche en s'appuyant sur les différents résultats obtenus.

VII.2 Environnement matériel et outils utilisés

VII.2.1 Véhicules, capteurs et environnement réels

La méthode présentée précédemment a été testée en environnement réel grâce une plateforme nommée Pavin (figure VII.1a) dédiée à la recherche sur les véhicules autonomes. Ce site expérimental comprend deux zones : une zone reproduisant un environnement urbain et une zone rurale, la totalité des deux zones fait environ 5000m^2. La zone urbaine est composée de rues avec différents types d'intersections, de marquage au sol et des feux tricolores fonctionnels. Des murs peints, de la végétation fournissent une apparence réaliste. En outre la zone entière est couverte par une station de base DGPS permettant la localisation RTK précise de tous les robots. Tous les tests sont menés sur des VipaLab (figure VII.1b) qui sont des véhicules électriques dédiés. Leurs petites dimensions (longueur 1.96 m, largeur 1.3 m et hauteur 2.11 m) sont appropriées pour des environnements urbains.

(a) plateforme Pavin présentant une zone urbaine et une zone rurale additionnelle.

(b) un VipaLab qui peut être équipé avec différents capteurs comme des télémètres, caméra, GPS, etc.

FIGURE VII.1 – Véhicules et environnement réel utilisés

Les tests de localisation ont été réalisés sur des véhicules VipaLab dont les capteurs

suivants ont été utilisés
- odomètre qui donne le mouvement relatif des roues ;
- capteur d'angle au volant ;
- télémètres Laser, SICK LMS151-10100, avec une portée jusqu'à 30 mètres, un angle de vue de -45 à 225 degrés, une résolution de 0.5 degrés ;
- GPS naturel avec une précision de l'ordre de 5 mètres ;
- GPS RTK, ProFlex 800, avec une précision centimétrique (utilisé uniquement pour l'évaluation des résultats).

Pour la communication entre les véhicules, utile notamment pour des expérimentations portant sur des convois, des modules wifi 802.11p sont utilisés.

VII.2.2 Simulateur Cobaye

Nous avons également à disposition un simulateur réaliste nommé Cobaye. Ce simulateur permet de créer ses propres scénarios mais fournit également la représentation d'environnements existants comme le site Pavin. Tout est réalisé de manière à avoir une simulation la plus réaliste possible. Ce simulateur est développé par l'entreprise 4D-Virtualiz [99].

(a) environnement de Pavin modélisé dans Cobaye

(b) véhicule modélisé dans Cobaye

FIGURE VII.2 – Modélisation de l'environnement réel à l'aide du simulateur Cobaye.

Ce simulateur accélère ainsi le développement en offrant une grande flexibilité (disponibilité, types de capteurs, nombre de véhicules, etc.). Les résultats déjà présentés dans les chapitres précédents, concernant en particulier la sélection de l'action et la mise à jour avec détection des erreurs, ont été réalisés en utilisant l'environnement fourni par le simulateur.

VII.2.3 Middleware utilisé : Effibox

L'accès aux données fournies par les capteurs est réalisé grâce à l'utilisation du logiciel Effibox développé par la société Effidence [6]. Ce logiciel permet en outre d'enregistrer des données afin de pouvoir les rejouer ultérieurement à la même cadence ou de façon accélérée ou ralentie, permettant ainsi un suivi et un débuggage de l'application optimaux. De plus, un système de listes d'observations, disponible avec ce middleware [141], permet d'intégrer de manière propre les différentes mesures capteur en prenant en compte la date d'observation et éliminant les problèmes consécutifs à la latence dans l'acquisition et le traitement des données.

VII.2.4 Bibliothèque de calcul de réseau bayésien

La bibliothèque utilisée pour réaliser les calculs liés au réseau bayésien est la bibliothèque dlib [76]. Cette bibliothèque, facile à mettre en place et à utiliser, permet de mettre rapidement en place un réseau bayésien. En revanche son utilisation doit se limiter à de petits réseaux en raison du temps de calcul qui devient prohibitif pour de trop grands réseaux. Dans notre cas, il serait possible d'améliorer le temps de calcul en développant un système prenant en compte l'aspect exclusivement binaire de nos événements et l'invariance temporelle de la structure de notre réseau. À titre d'exemple, la construction et l'utilisation d'un réseau bayésien, avec cette librairie, nécessite un temps de calcul de 10 à 15 ms pour calculer la probabilité de bonne détection d'un triplet.

VII.2.5 Architecture de calcul

L'ordinateur utilisé sur les véhicules pour des applications réelles possède un processeur Intel i686 avec 4 cœurs et cadencé à 1.6 GHz.

VII.2.6 Système d'Information Géographique

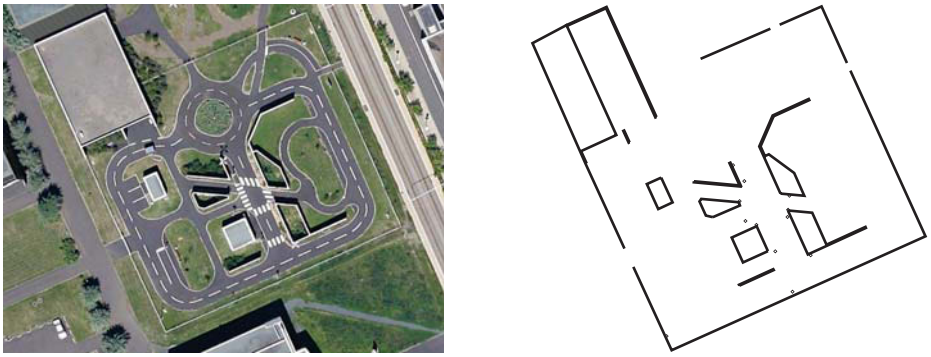


FIGURE VII.3 – A gauche : image du site Pavin (Google Maps). A droite : modélisation de cet environnement dans la base de données avec la représentation du grillage, des trottoirs, des murs et des poteaux

Afin de gérer de manière efficace les objets spatiaux comme les murs, les bâtiments, les trottoirs, etc. du site expérimental, nous avons fait le choix d'utiliser un Système d'Information Géographique (SIG). Pour cela nous avons utilisé le système de gestion de base de données « PostGis » qui est une extension du langage PostGreSQL. PostGis rajoute la prise en compte d'objets localisés géographiquement [123]. Les murs et les poteaux présents dans l'environnement sont présents dans notre SIG. La figure VII.3 montre une vue de notre environnement Pavin sur Google Maps et la modélisation à l'aide du SIG.

VII.2.7 Observabilité des triplets et champ de vue capteur

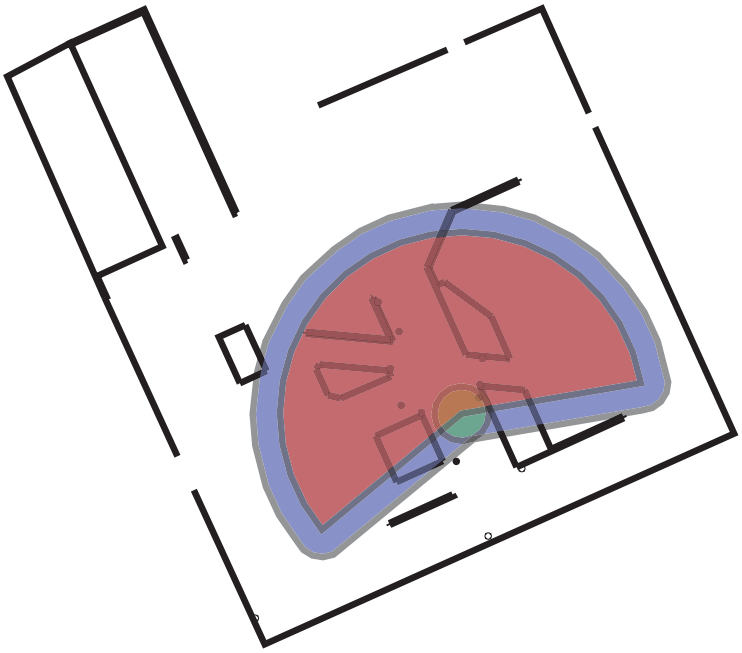
Précalcul de la zone d'observabilité des triplets

La zone d'observabilité de chaque triplet (voir section V.3.1 page 82) est calculée hors ligne et accessible directement dans la base de données du SIG. Ceci permet

d'éviter un temps de calcul non négligeable.

Détermination du champ de vue du capteur

(a) Zone de visibilité idéale



(b) Zone de visibilité avec prise en compte de l'incertitude

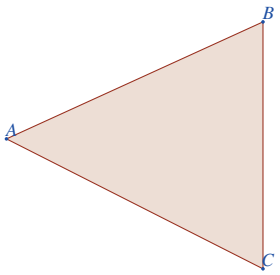
FIGURE VII.4 – Zone de visibilité d'un télémètre laser situé au point T

A partir de l'estimation de la position du robot, la position des capteurs est déduite

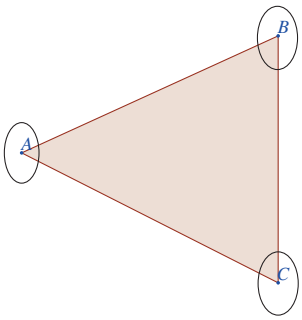
immédiatement. Pour chaque capteur, une zone de visibilité est calculée déterminant ainsi quels amers pourront être vus avec ce capteur. La figure VII.4a montre la zone de visibilité pour un télémètre laser. L'utilisation conjointe de cette zone avec le SIG permet à chaque instant de connaître les amers potentiellement visibles par le robot et donc les différents triplets perceptifs disponibles.

Intégration de l'incertitude capteur

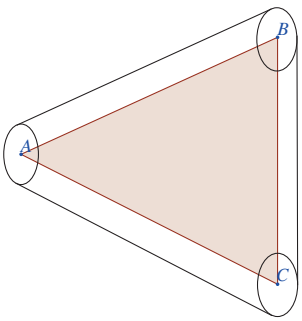
Dans la pratique, la position du robot n'est pas connue de manière précise, mais avec une certaine incertitude, et par conséquent la position du capteur non plus. Il est donc nécessaire de prendre en compte cette incertitude. En réalité, le robot peut être partout dans la zone d'incertitude, aussi la zone de sélection des triplets devra inclure toutes les zones possibles. Pour réaliser cela, la zone de visibilité de base de l'amer va être « dilatée » par l'incertitude de positionnement du capteur (voir figure VII.4b). Le principe global de l'algorithme de dilatation est le suivant : en chaque point constituant le polygone représentant la zone de visibilité du capteur est projetée l'incertitude associée à la position estimée du capteur, cette incertitude sera donc représentée par une ellipse. La zone de visibilité finale sera alors le polygone convexe englobant les zones centrées sur chaque point du polygone. Les différentes étapes sont visibles à travers un exemple sur la figure VII.5



(a) Zone de visibilité du capteur initiale (ici un triangle)



(b) Projection de l'incertitude du capteur sur les différents points du polygone.



(c) Construction du polygone convexe englobant qui sera la nouvelle zone de visibilité.

FIGURE VII.5 – Illustration de l'intégration de l'incertitude capteur.

VII.3 Résultats

Différents résultats obtenus avec plusieurs types de capteurs (un télémètre, plusieurs télémètres, un capteur d'angle, un capteur GPS, ...) sont présentés ici. Les situations peuvent être diverses suivant le principe illustré mais nous distinguons deux grandes parties : la localisation statique c'est-à-dire que le véhicule est à l'arrêt, la localisation dynamique durant laquelle le véhicule peut être en mouvement, soit conduit manuellement, soit conduit automatiquement. Des tests statistiques sont également effectués permettant d'appréhender la robustesse de l'algorithme. Enfin des applications plus complètes et permettant d'illustrer l'utilisation de notre système de localisation dans le cadre d'applications de navigation automatique sont présentées.

VII.3.1 Préliminaires

Incertitude fournie sur la position

Pour la présentation des résultats, l'incertitude dont il sera question sera toujours à comprendre comme étant l'incertitude à un écart-type.

Méthode de calcul de l'erreur de localisation et interprétation des résultats

L'erreur de localisation au cours du temps, qui est montrée lors de la localisation dynamique, est calculée en appliquant les étapes suivantes :

- À chaque position fournie par le récepteur du GPS RTK cadencé à 10Hz, le système stocke la date de cette information ;
- à la prochaine position estimée par notre système, ce dernier interpole les valeurs dont les dates correspondantes encadrent celle précédemment stockée du GPS RTK ;
- la valeur absolue de la distance entre la position interpolée et la position fournie par le récepteur GPS RTK est alors donnée comme étant l'erreur de localisation.

Cette erreur est calculée en considérant que la vitesse du véhicule est constante (ce qui est tout à fait valable au vu des vitesses des véhicules et de la faible distance durant laquelle cette approximation est réalisée).

Cette erreur donnée peut ne pas être pertinente en raison d'erreurs entachant la position estimée par le récepteur GPS RTK. Deux principales causes sont présentes :

- Le véhicule passe à l'intérieur d'un bâtiment, le signal GPS est alors faux ;
- le véhicule est dans une pente assez forte. L'antenne réceptrice étant sur le toit du véhicule, il y a un effet de levier qui peut engendrer une fausse estimation de la position allant jusqu'à une dizaine de centimètres sur le site de PAVIN.

VII.3.2 Localisation statique

Dans un premier temps, la localisation statique est étudiée, c'est-à-dire que le véhicule est à l'arrêt. La position estimée est fournie initialement avec une incertitude plus ou moins grande suivant les situations. Cette incertitude peut porter sur la position métrique ou sur l'orientation du véhicule. La position initiale est toujours fournie avec une incertitude suffisamment grande pour que cette position soit intègre. La confiance initiale est égale à 1 c'est-à-dire que le système sait que sa position initiale est forcément intègre.

Évolution de l'algorithme avec une incertitude de départ faible

La situation étudiée ici (figure VII.7) est la suivante : le véhicule est situé au départ avec une position estimée fiable et avec une incertitude de 2 mètres (l'incertitude est représentée par le cercle noir autour du robot) et une orientation initiale précise à 10° près. Le système va alors sélectionner au fur et à mesure les triplets qui semblent les plus pertinents afin d'atteindre son objectif final de précision et de confiance. L'objectif de précision est fixé à 15 centimètres pour la position (x, y) et 2 degrés pour l'angle. L'objectif de confiance est de 99%.

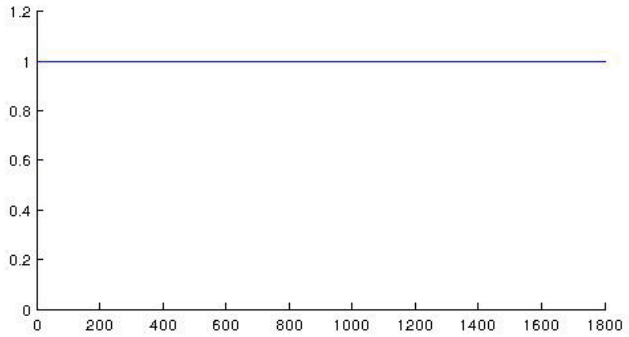
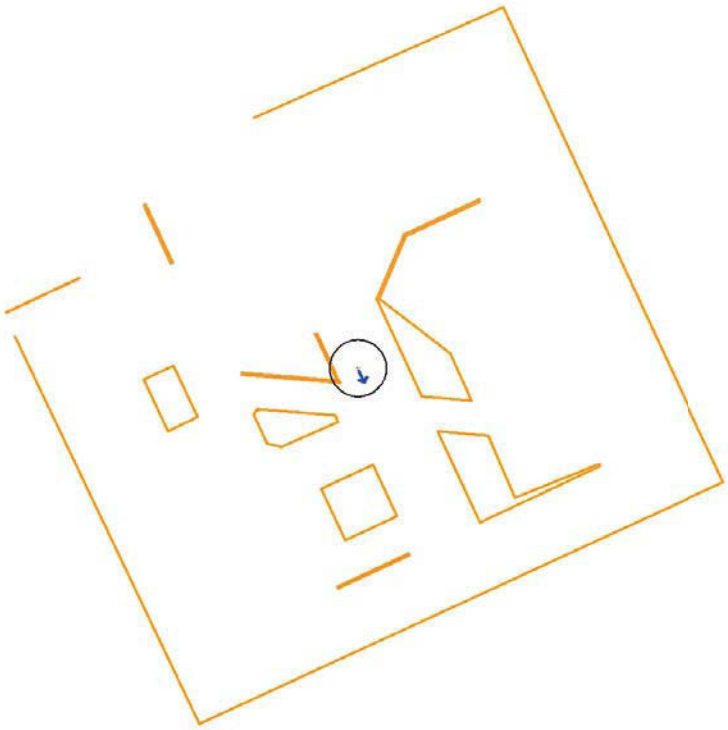


FIGURE VII.6 – Localisation statique, estimation de la position du robot à l'aide d'un télémètre laser, l'incertitude départ est faible (2 mètres) : évolution de la confiance au cours du temps (ms). À chaque instant, le système sélectionne des triplets qui sont faciles à détecter, sans ambiguïté et avec une très faible probabiltié de réaliser une mauvaise association, la confiance, réévaluée à chaque fois, est donc toujours supérieure à 0.99.

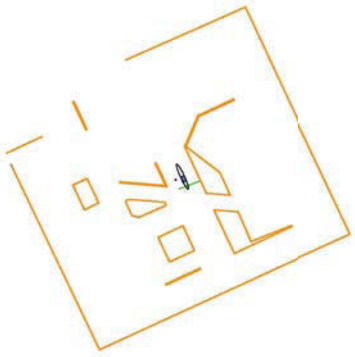
L'objectif de précision est atteint en quatre étapes. Les deux premières étapes sont montrées sur la figure VII.7. A chaque instant l'amer faisant partie du triplet sélectionné est identifiable grâce au trait vert tiré entre le véhicule et l'amer. À la première étape (figure VII.7b), c'est un mur situé à la gauche du véhicule qui est sélectionné et détecté permettant ainsi une première mise à jour qui réduit l'incertitude selon la direction perpendiculaire au mur. À la deuxième étape (figure VII.7c), un autre mur est sélectionné, ce mur est dirigé perpendiculairement au premier et permet ainsi de réduire l'incertitude dans la direction perpendiculaire à la précédente. L'incertitude est grandement réduite, c'est d'ailleurs pour cette raison que ce mur a été sélectionné par le système. Deux autres étapes de détection sont alors nécessaires pour atteindre la précision demandée. Sans surprise, la confiance reste supérieure à 99% tout au long de ces étapes (figure VII.6), en effet l'incertitude est suffisamment faible pour que le système sélectionne à chaque instant des triplets qui sont facilement détectables et sans ambiguïté possible.

Évolution de l'algorithme avec une grande incertitude de départ et un récepteur GPS disponible

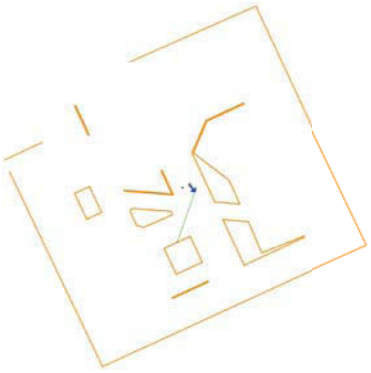
Nous avons ici le même scénario que précédemment mais avec un autre capteur disponible : un GPS permettant de fournir une position estimée avec une incertitude d'environ 3 mètres. Plutôt que de fournir initialement une position de départ avec une



(a) Position initiale



(b) Première étape



(c) Deuxième étape

FIGURE VII.7 – Localisation statique, estimation de la position du robot à l'aide d'un télémètre laser, l'incertitude de départ est faible (2 mètres) : évolution de l'estimation. En orange : amers repertoriés, ellipse : espace d'incertitude de la position estimée.

incertitude de 5 mètres, une première estimation de la position est donnée avec une incertitude d'environ 16 mètres. A la première itération, notre système va immédiatement choisir le GPS comme capteur le plus pertinent. En effet, le véhicule est situé en extérieur ce qui implique que le GPS est disponible. La probabilité de réaliser une bonne détection est donc extrêmement forte. Le GPS n'a pas encore été utilisé puisque

nous sommes à la première itération, il n'y a donc pas de problème de corrélation de données. Tout ceci fait que le GPS est choisi par notre système et va lui fournir une position estimée avec une incertitude beaucoup plus faible. Après cette première étape, nous nous retrouvons alors dans la situation décrite précédemment sur la figure VII.7. Le GPS n'est plus sélectionné puisqu'il fournirait une information corrélée avec celle déjà exploitée. L'algorithme converge alors en seulement 5 étapes.

Cette mise en situation montre ici la pertinence de l'approche Top-Down développée qui permet de converger efficacement vers la solution en choisissant le meilleur triplet à chaque instant.

Évolution de l'algorithme avec une grande incertitude de départ et sans récepteur GPS

La situation présentée ici (figure VII.9) est la même que précédemment mais sans récepteur GPS disponible. Cette grande incertitude implique pour le système un nombre conséquent de triplets disponibles ce qui conduit à un temps de calcul plus important lors de la phase de sélection, des zones de focalisation peu discriminantes, une plus grande probabilité de tomber sur un minimum local. Les objectifs de précision et de confiance restent les mêmes.

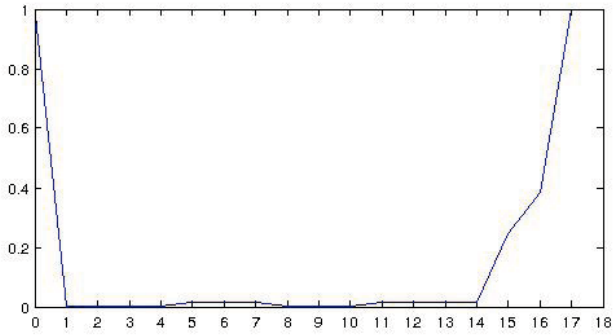
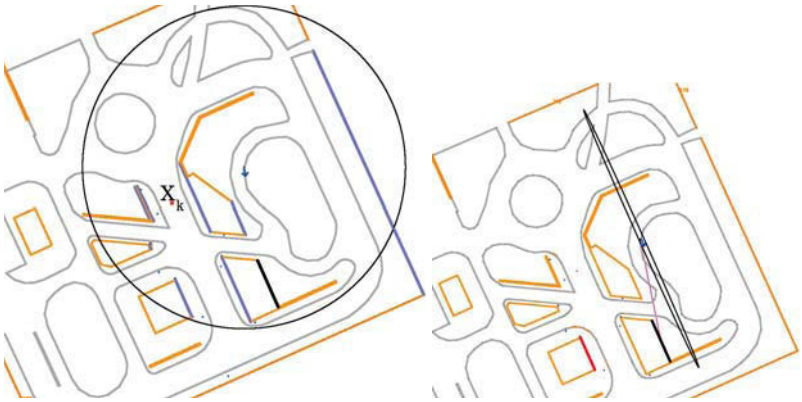
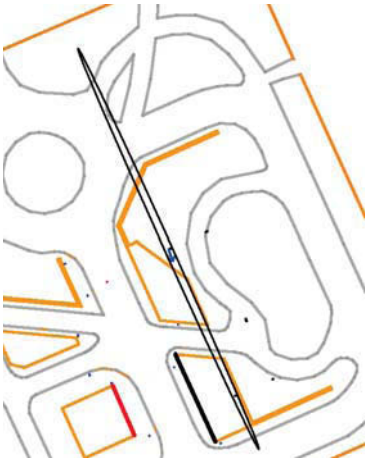


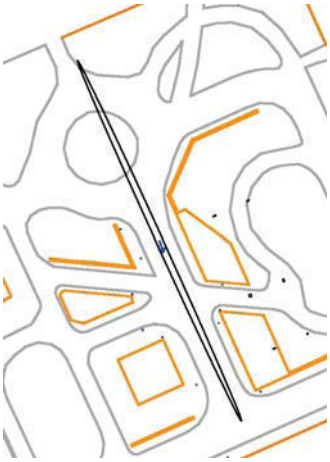
FIGURE VII.8 – Localisation statique, estimation de la position du robot à l'aide d'un télémètre laser, l'incertitude de départ est élevée (16 mètres) : évolution de la confiance à chaque itération.



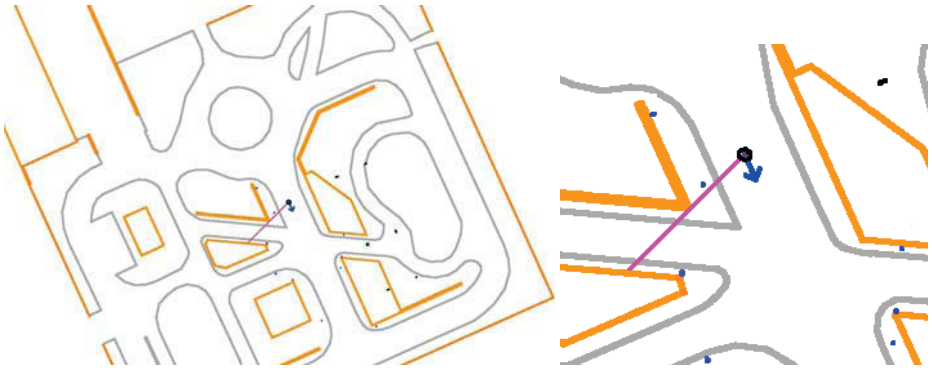
(a) Sélection du premier triplet, l'amer à détecter est l'amer en noir, il y a un grand risque de confusion puisque 8 autres amers sont compatibles (en bleu sur l'image de gauche). L'amer qui est effectivement détecté est le rouge sur l'image de droite mais il est associé avec le noir ce qui vient donner une estimation fausse de la position.



(b) Mise à jour 11 : re-association de la première observation avec le premier voisin soit l'amer en noir, la nouvelle position estimée ne correspond toujours pas à la position réelle.



(c) Mise à jour 13 : re-association de la première observation avec l'amer adéquat, la position estimée est maintenant intègre mais non précise et avec une confiance très basse (0.01).



(d) Étape 17 : la position estimée est intègre avec une bonne précision (moins de 10 centimètres) et une bonne confiance(0.99).

FIGURE VII.9 – Localisation statique, estimation de la position du robot à l'aide d'un télémètre laser, l'incertitude de départ est élevée (16 mètres) : évolution de l'estimation, 17 étapes sont nécessaires pour atteindre les objectifs. En orange : amers repertoriés, ellipse : espace d'incertitude de la position estimée.

Au départ, le système évalue un amer (en noir sur la figure VII.9a) comme étant le plus pertinent et essaie donc de le détecter (figure VII.9a). Cependant en raison de la grande incertitude de la position initiale, la zone de focalisation est très importante et n'est d'aucun secours pour nous aider à améliorer l'association de données. Le système va alors échouer à détecter les deux triplets suivants et va en déduire que sa première association n'était pas bonne, il va donc la remettre en cause et réassocier l'observation initiale. Les voisins compatibles étaient nombreux (8) aussi le système va devoir réitérer ce processus avant de réaliser enfin la bonne association à l'étape 13 (figure VII.9c), à partir de là il va arriver à détecter correctement les amers suivants permettant à la fois d'améliorer sa précision et sa confiance jusqu'à atteindre une précision et une confiance correctes à l'étape 17 (figure VII.9d).

La figure VII.8 montre que la confiance est très basse tout au long du processus et remonte seulement sur la fin quand la position estimée est correcte et qu'à chaque itération la nouvelle détection d'un amer vient lever les diverses ambiguïtés. Ce comportement est celui attendu.

Cette mise en situation montre ici l'utilité et la performance du système de remise en cause.

Problème du robot kidnappé

Nous appelons ici problème du robot kidnappé la situation où le robot ne sait absolument pas où il se trouve dans son environnement. La seule donnée dont dispose le robot c'est qu'il se trouve dans la carte qui lui est fournie. Ce problème est bien connu et représente un challenge important [78, 97].Nous pouvons donc considérer que nous nous trouvons dans la situation du robot kidnappé quand notre espace d'incertitude englobe quasiment notre carte. Dans un premier temps, nous verrons quels sont les cas d'échecs de l'algorithme, c'est-à-dire dans quelles situations le processus ne converge pas. Nous verrons enfin les statistiques sur la convergence de notre algorithme vers la bonne position suivant l'incertitude initiale sur la position ou sur l'angle.

Cas d'échec de l'algorithme Il peut arriver que le système ne converge pas vers la bonne position. Cette situation peut survenir quand la position estimée du robot est fausse et que le système ne peut plus sélectionner de triplets lui permettant de détecter une anomalie et de déclencher le processus de remise en cause. Afin d'illustrer ce concept, considérons la figure VII.10.

Dans ce cas précis, l'algorithme croit avoir détecté les deux murs en noir (alors qu'en réalité il a détecté les murs en bleu), dans cette situation aucun autre amer n'est disponible pour lui puisqu'ils sont tous cachés par les deux murs précités. L'algorithme ne cherche donc aucun autre amer et s'arrête là, cet état est appelé état-puits. La situation ne pourra être débloquée que si d'autres amers deviennent disponibles, tant que le véhicule reste statique cela ne sera donc pas possible. Ce cas de figure met en avant la nécessité d'avoir un nombre d'amers suffisants pour lever une ambiguïté. Cependant si une application de guidage automatique utilise cet algorithme de localisation, ce dernier lui fournit l'estimation de la position qui est fausse mais également un niveau de confiance qui en l'occurrence sera bas car si cette situation est arrivée cela est généralement dû au fait que le triplet recherché avait des voisins, ce qui est pris en compte dans l'application. Le système de navigation saura alors que la position estimée est très probablement fausse et ne peut donc être exploitable. Plus le nombre de triplets disponibles est grand (grâce à la richesse de l'environnement, le nombre de détecteurs

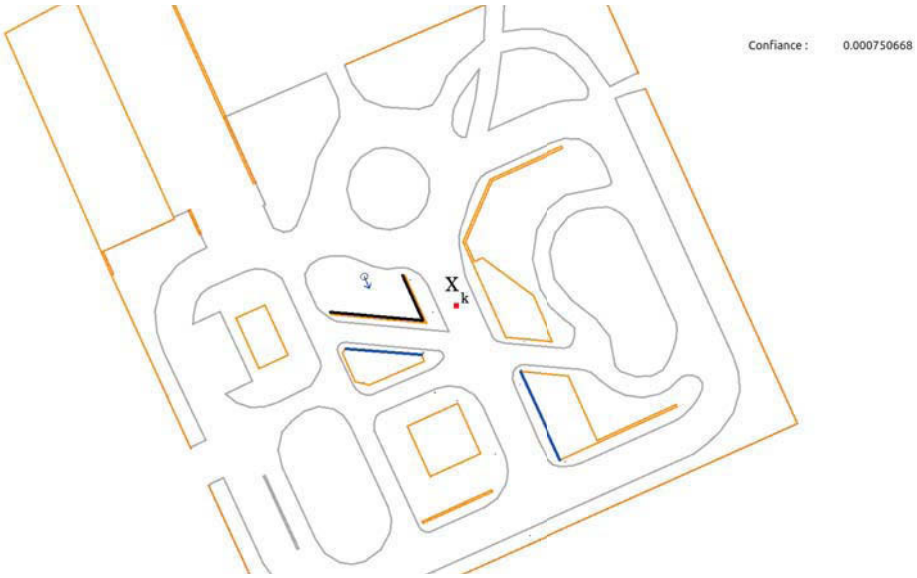
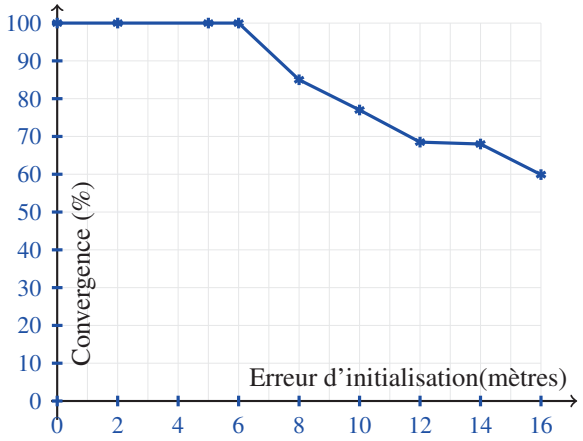


FIGURE VII.10 – Localisation statique, estimation de la position du robot à l'aide d'un télémètre laser, l'incertitude de départ est élevée (16 mètres) : convergence vers un état-puits. Le robot a essayé de détecter les deux murs en noir sur l'image mais en réalité a détecté et donc mal associées les deux détections. La nouvelle position estimée est fausse mais dans cette position aucun nouveau triplet n'est disponible, ainsi bien que la confiance soit très faible (moins de 0.001 %) le système est donc bloqué. Cet état est appelé état-puits.

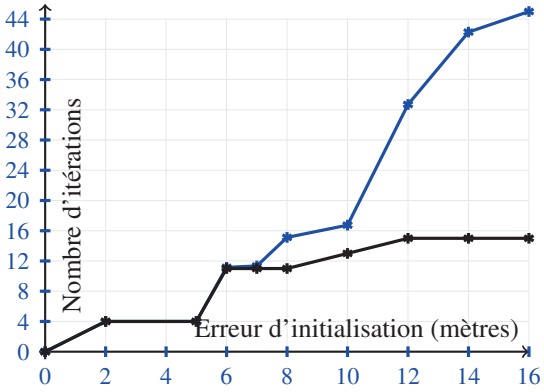
disponibles ou le nombre de capteurs fournis), plus la probabilité de se trouver dans une telle situation est faible.

Statistiques Nous avons vu précédemment qu'à partir d'une incertitude très grande (16 mètres) qui englobe presque toute la carte, il était possible au système de réaliser une estimation vraie de la position du robot et d'atteindre ses objectifs de précision et de confiance. Cependant, nous avons également vu qu'il existait des cas de figure où l'algorithme ne convergeait pas vers une position correcte. Afin de connaître plus précisément, les caractéristiques de notre système dans notre environnement, l'algorithme a été évalué avec différentes valeurs d'initialisation. Nous pouvons voir les résultats sur les figures VII.11 et VII.12a. Pour cette évaluation, l'algorithme est répété 500 fois, avec des tirages aléatoires, pour chaque point de la courbe, c'est-à-dire pour chaque erreur initiale. Sur la figure VII.11, les erreurs d'initialisation sont des erreurs de translation uniquement, l'orientation est donnée à 5° près.

En-dessous de 6 mètres d'erreur initiale, le système converge systématiquement vers la bonne position. Au delà, les performances du système baissent jusqu'à 60 % de réussite pour 16 mètres d'incertitude initiale. Les tests n'ont pas été effectués avec une erreur supérieure à 16 mètres car cela risquerait de donner une position initiale en dehors de la carte. Dans chaque cas où l'algorithme ne converge pas la confiance est inférieure à 90%, ce qui signifie que dans chacun de ces cas le système sait qu'il ne peut se fier totalement à la position finale estimée. La figure VII.11b montre le nombre d'itérations nécessaire pour atteindre l'objectif lorsque ce dernier est atteint. En-



(a) Pourcentage de convergence



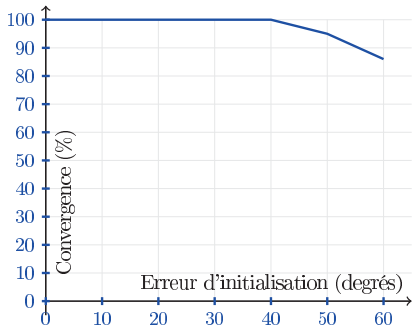
(b) Nombre d'itérations nécessaires pour atteindre l'objectif dans les cas de convergence. En noir : le nombre médian d'itérations, en bleu : le nombre moyen d'itérations

FIGURE VII.11 – Localisation statique : performances de l'algorithme avec différentes erreurs initiales de translation

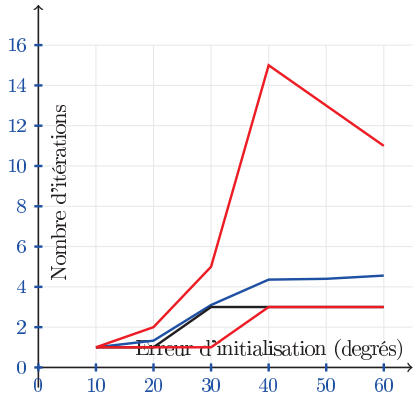
dessous de 6 mètres d'erreur initiale, l'objectif est atteint en 4 itérations en moyenne. Au delà, le nombre d'itérations moyen grandit très vite pour monter jusqu'à 44 itérations pour une erreur de 16 mètres. Toutefois, ce nombre est grandement dépendant de la position estimée initiale du robot, suivant les types d'amers à proximité de cette position, les configurations identiques, etc. Le nombre d'itérations pour chaque valeur de l'erreur initiale ne suit absolument pas une courbe gaussienne, nous avons donc affiché également le nombre médian d'itérations qui est beaucoup plus faible. Pour une erreur initiale de 16 mètres, le nombre médian d'itérations est de 15. Ainsi si le nombre moyen est de 44, la moitié des essais convergent en moins de 15 itérations. Au vu de cette répartition, l'écart-type de ces itérations ne signifie rien et n'est donc pas donné ici.

Sur la figure VII.12a, l'erreur initiale ne porte plus sur la translation (la position de départ est correcte à 2 centimètres près) mais sur l'orientation. L'algorithme est considéré comme ayant convergé vers la bonne solution quand la position estimée finale est à

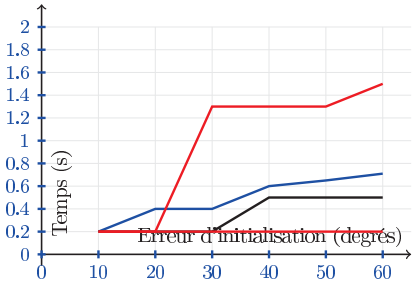
moins de 2 centimètres de la position réelle, que l'orientation est correcte à 2° près et que la confiance est à un niveau supérieur à 0.99.



(a) Pourcentage de convergence



(b) Nombre d'itérations. En rouge les extrema, en bleu le nombre moyen d'itérations, en noir le nombre médian d'itérations



(c) Temps de calcul. En rouge les extrema, en bleu le nombre moyen d'itérations, en noir le nombre médian d'itérations

FIGURE VII.12 – Localisation statique : performances de l'algorithme avec différentes erreurs initiales sur l'orientation

Tant que l'orientation initiale a une erreur inférieure à 40 degrés, l'algorithme converge dans 100% des cas. Au delà, il y a un risque que l'algorithme ne converge pas. Cette fois, cela est rarement dû à la présence d'un état-puits (contrairement aux situations de non-convergence dans le cas d'une erreur initiale de translation) mais plutôt à un problème de non-linéarité engendrant une divergence du filtre de Kalman lors de la mise à jour. La figure VII.12b montre le nombre d'itérations nécessaire pour converger. Contrairement au cas d'une erreur initiale de translation, l'algorithme ne dépasse jamais les 15 itérations et généralement le nombre de ces dernières se situe plutôt autour de 4 itérations en moyenne. En effet, il y a beaucoup de configurations identiques, l'algorithme doit donc rarement se remettre en cause. Généralement quand l'algorithme échoue, cela est dû à un problème de non-linéarité engendrant une mauvaise mise à jour du filtre de Kalman comme nous l'avons déjà évoqué. À titre d'exemple, la figure VII.12c montre le temps de calcul nécessaire. Plus l'erreur initiale est importante, plus le nombre de triplets potentiels est important et donc plus le temps de calcul augmente. Les valeurs des temps de calcul ne sont pas très significatives en elles-mêmes

car l'architecture générale du programme et la programmation pourraient largement être améliorées. Des travaux encore en cours montrent, par exemple, qu'une implémentation plus performante (réalisée dans l'équipe mais non exploitée ici) des réseaux bayésiens permet de réduire le temps de calcul d'une inférence par 15.

Conclusion sur le robot kidnappé Le système développé est très performant à la condition expresse de ne pas tomber dans ce que nous appellerons un état-puits, c'est à dire une estimation de position qui est fausse mais dans laquelle aucun nouvel amer ne peut être sélectionné (soit parce que le système estime qu'il n'y a pas d'amer à voir, soit parce qu'ils ont déjà tous été essayés et fournissent donc une information corrélée). Cependant, dans ces cas-là le système sait généralement que la position estimée a une faible probabilité d'être intègre. Dans le cas d'une conduite autonome, le programme sera donc capable de dire qu'il n'est pas dans les bonnes conditions pour piloter le véhicule de manière autonome. Dans tous les cas, le comportement est celui attendu, la solution pour résoudre ce problème serait d'étendre les nombre de triplets perceptifs.

VII.3.3 Localisation dynamique

Nous avons vu précédemment comment le problème de l'initialisation pouvait être résolu grâce à notre algorithme. Nous appliquons maintenant ce même algorithme mais avec un véhicule en mouvement. Sauf précision contraire, l'initialisation métrique au départ est correcte à 5 mètres près environ, l'initialisation angulaire à 10° près et la confiance initiale est de 1. Les véhicules VipaLab (cf figure VII.1b) circulent à une vitesse comprise entre 5 et 10 km/h. Nous montrons ici les résultats obtenus avec différents types de capteurs et associations de capteurs.

Dans un premier temps, la localisation à l'estime seule est montrée, sans utiliser donc notre système Top-Down. Tous les autres résultats présentés ensuite sont présentés avec différents capteurs mais en utilisant toujours la localisation à l'estime (dans le filtre de Kalman, la localisation à l'estime permet de réaliser la phase de prédiction tandis que les autres capteurs permettent de réaliser les phases de mise à jour).

Localisation à l'estime

On appelle localisation à l'estime, la technique d'estimation de la position du robot se basant sur les mesures des déplacements élémentaires effectués depuis la dernière estimation. Les capteurs privilégiés pour ce type de localisation sont les capteurs proprioceptifs comme les odomètres, les centrales inertielles, etc. Dans notre cas, nous utilisons ce système comme phase de prédiction dans le filtre de Kalman. Le modèle d'évolution du véhicule est supposé assimilé au modèle bicyclette, ce qui est courant dans ce genre d'application.

La figure VII.13 montre le résultat de la localisation lorsque seule la localisation à l'estime, à l'aide d'un odomètre et d'un capteur d'angle au volant, est utilisée. La forme globale de la trajectoire est la même que celle de la trajectoire de référence mais on peut remarquer qu'il y a une dérive au cours du temps, ce qui est normal compte tenu de la méthode utilisée (à chaque instant, on a une accumulation d'erreurs qui s'additionnent aux précédentes). Quand le véhicule passe dans le garage, le récepteur GPS ne reçoit pas de signal correct, ce qui explique que la trajectoire de référence (fournie par le récepteur GPS RTK) n'est pas disponible à cet endroit du trajet.

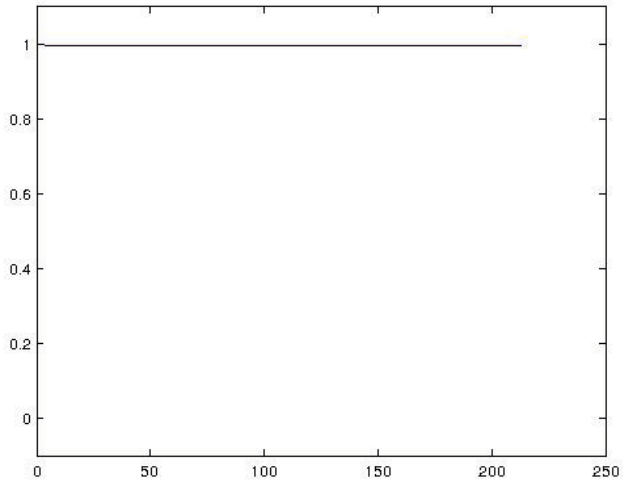
L'odomètre et le capteur d'angle au volant seront toujours utilisés par la suite en combinaison avec les autres capteurs.

FIGURE VII.13 – Localisation dynamique réalisée à l'estime avec un odomètre et un capteur d'angle au volant. En noir : la trajectoire estimée, en cyan : la trajectoire fournie par le récepteur GPS RTK. Le véhicule représenté en jaune se trouve au point d'arrivée, le point de départ se trouve au milieu du carrefour (près des passages piétons). Le véhicule passe dans le garage, occultant le signal GPS, aussi à cet endroit la trajectoire de référence n'est pas disponible.

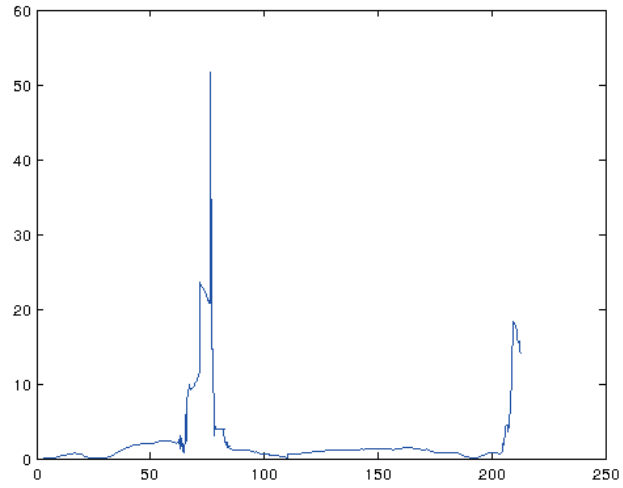
Localisation avec un capteur d'angle

Le capteur rajouté ici est un capteur d'angle (en plus du capteur d'angle au volant), c'est-à-dire qu'il nous fournit l'orientation du véhicule par rapport au nord. Cette fonction peut être réalisée par exemple par une magnétomètre qui indiquerait la position par rapport au nord magnétique. N'ayant pas de magnétomètre disponible, nous avons simulé un capteur d'angle grâce à l'orientation disponible dans les trames fournies par le récepteur GPS Ublox. Nous avons ainsi accès à l'angle du véhicule quand ce dernier est en mouvement et que le GPS est disponible. Si le véhicule est à l'arrêt ou à l'intérieur d'un bâtiment, l'angle fourni sera incorrect. Ce capteur sera considéré par le système au travers d'un triplet perceptif (contrairement au capteur d'angle au volant) constitué du capteur d'angle comme capteur, d'un détecteur très simple (l'identité puisque l'angle étant directement fourni par le capteur, aucun traitement n'est donc nécessaire), et l'amer est inexistant. La localisation réalisée à l'estime, vue précédemment, est toujours utilisée lors de la phase de prédiction du filtre de Kalman, les données fournies par les triplets sélectionnés sont utilisées pour la phase de mise à jour. Pour calculer l'occultation possible de ce capteur, nous avons utilisé la méthode de calcul développée pour le GPS (voir V.3.1).

Sur la figure VII.15, il est possible de voir que le capteur d'angle est correctement intégré et pris en compte permettant d'améliorer l'estimation de la trajectoire. Toute-



(a) Évolution de la confiance au cours du temps



(b) Erreur de localisation (m) au cours du temps (s)

FIGURE VII.14 – Localisation dynamique, estimation de la trajectoire avec un capteur d'angle : évolution de la confiance et de l'erreur au cours du temps

fois, le problème de la dérive subsiste même s'il est réduit.

Nous pouvons également remarquer deux grands pics d'erreur (figure VII.14b). Ces pics se produisent car le véhicule est à l'intérieur du bâtiment et la trajectoire de référence, qui est fournie par un GPS RTK, est alors fausse, ce qui explique un si grand écart. Si on enlève ces passages, l'erreur moyenne est de 1 mètre avec un écart-type de 60 cm. La précision obtenue n'est pas suffisante pour un guidage automatique des véhicules. Le graphe de la confiance (figure VII.14a) reste toujours stable et supérieur

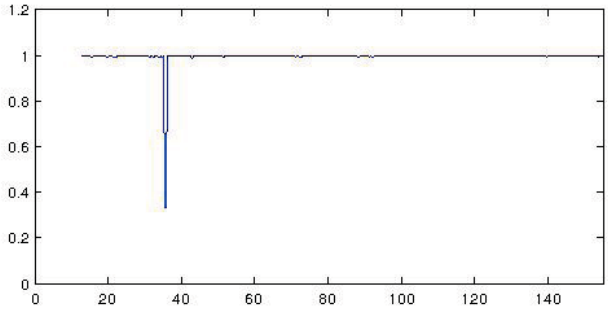
(a) Trajectoire effectuée. En noir : la trajectoire estimée, en cyan : la trajectoire fournie par le récepteur GPS RTK.

FIGURE VII.15 – Localisation dynamique, estimation de la trajectoire avec un capteur d'angle : évolution de l'algorithme. En noir, la trajectoire estimée. En bleu, la trajectoire de référence fournie par le récepteur GPS RTK. Le véhicule passe dans le garage, occultant le signal GPS, aussi à cet endroit la trajectoire de référence, fournie par le récepteur GPS RTK, est très éloignée de la réalité, provoquant la présence des grands pics visibles sur le graphe.

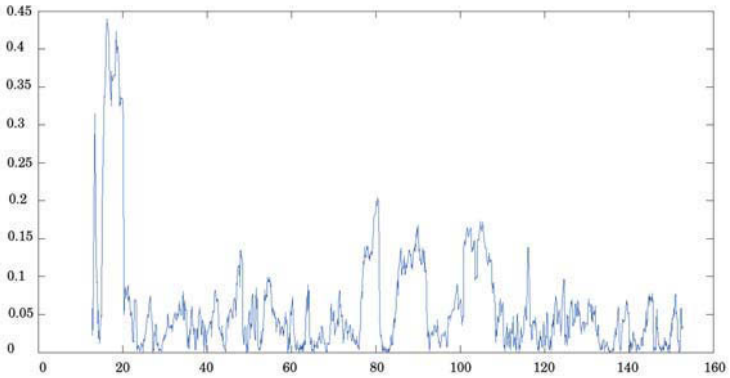
à 0.99, en effet en extérieur ce capteur est toujours disponible, il n'y a pas d'ambiguïté possible. Il peut y avoir un problème avec ce capteur lorsque le véhicule est à l'intérieur d'un bâtiment, en effet le signal GPS est faussé à l'intérieur, mais ceci est pris en compte par notre système qui ne sélectionne pas ce capteur dans cette situation.

Localisation avec un télémètre

Nous nous plaçons ici avec un seul capteur, un télémètre laser situé à l'avant du véhicule (le capteur d'angle de la section précédente n'est donc pas utilisé ici). Considérons la figure VII.17 qui montre le résultat sur une trajectoire d'environ 200 mètres.



(a) Évolution de la confiance au cours du temps (s)



(b) Erreur de localisation (m) au cours du temps (s)

FIGURE VII.16 – Localisation dynamique, estimation de la trajectoire avec un télémètre laser : évolution de la confiance et de l'erreur au cours du temps (s)

La figure VII.17 montre l'ensemble de la trajectoire parcourue par le véhicule avec la trajectoire fournie par le GPS RTK, qui est la trajectoire de référence, et la trajectoire estimée par notre système de localisation. Les amers utilisés sont ici les murs et le grillage entourant le site Pavin. D'un point de vue purement qualitatif, nous pouvons déjà voir que les deux trajectoires (estimée et fournie par le GPS RTK) sont assez proches. La sous-figure VII.16b montre plus précisément dans quelle mesure les deux courbes sont éloignées en donnant l'erreur au cours du temps de la trajectoire estimée par rapport à celle de référence. L'erreur moyenne de la trajectoire est de 6 centimètres avec un pic à 44 centimètres qui survient au début, le temps que le système arrive à réduire son incertitude. Après cela, l'erreur reste en général inférieure à une dizaine de centimètres. Nous pouvons toutefois observer plusieurs pics au-dessus de 15 centimètres à partir de 75 secondes. Ces pics correspondent aux moments où le véhicule n'arrive pas à voir d'amers et ne peut donc se mettre à jour. Le premier pic survient car le véhicule se trouve à un endroit d'où il est censé pouvoir voir des amers mais ceux-ci ne sont pas visibles à cause de la pente assez forte du terrain à cet endroit. Ceci a pour

FIGURE VII.17 – Localisation dynamique, estimation de la trajectoire avec un télémètre laser : évolution de l'algorithme. En noir, la trajectoire estimée. En bleu, la trajectoire fournie par le GPS RTK.

conséquence directe que le faisceau laser émis par le télémètre ne peut atteindre ces amers, étant « arrêté » par la pente, ceci est illustré sur la figure VII.18.

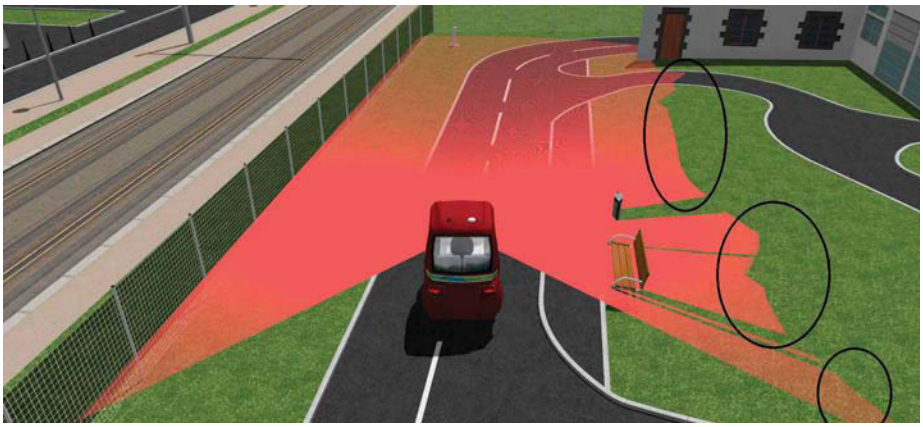


FIGURE VII.18 – Illustration des rayons du télémètre laser arrêtés par la pente aux endroits entourés par des ellipses noires.

Le système navigue alors à l'odométrie, ce qui implique une dérive. Les deux autres pics correspondent à des endroits où peu d'amers sont visibles par le véhicule et sont

dus, eux aussi, à la dérive liée à l'odométrie. Sur la courbe de la confiance, nous voyons seulement une chute de cette dernière vers 38 secondes. Cela correspond à une non-détection d'un mur à cause d'un problème de détecteur immédiatement suivie de la non-détection d'un autre mur car ce dernier est caché par des buissons non-répertoriés.

Une vidéo sur la localisation d'un véhicule avec un télémètre montrant la trajectoire estimée au cours du temps ainsi que la focalisation dans l'espace capteur est disponible sur internet à l'adresse suivante : `https://youtu.be/a_fN8Ae3S7A`.

Localisation avec un télémètre et un capteur d'angle

Nous avons maintenant intégré un télémètre laser et le capteur d'angle absolu déjà utilisé seul dans le système (voir section VII.3.3). Cela signifie en particulier que le système doit sélectionner à chaque instant des triplets concernant soit le télémètre soit le capteur d'angle. Tout ceci est géré de manière totalement transparente par notre algorithme. La trajectoire du véhicule utilisée dans notre exemple (figure VII.20) fait environ 380 mètres.

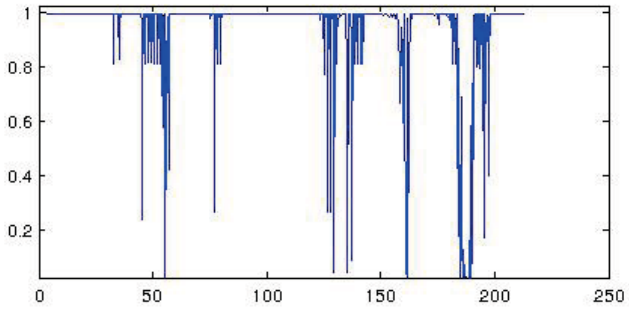


FIGURE VII.19 – Localisation dynamique, estimation de la trajectoire avec un télémètre et un capteur d'angle : évolution de la confiance au cours du temps (s).

Les résultats montrent effectivement une amélioration, spécialement dans le cas où notre véhicule s'engage dans une partie ne possédant aucun amer détectable avec notre télémètre laser. Nous avons ici réalisé une trajectoire plus grande que les autres puisque nous sommes sortis de l'environnement urbain. Sur cette trajectoire, nous pouvons remarquer plusieurs points marquants entourés d'une ellipse rouge et numérotés, nous les détaillons ici :

1. À cet endroit, nous pouvons distinguer un saut dans la trajectoire. Cela est dû au fait que les amers visibles auparavant ne fournissaient que des informations suivant une seule direction. À partir de cet instant, un nouvel amer est disponible donnant une information suivant une autre direction et permet au véhicule de se recaler, ce qui provoque ce saut. L'ellipse d'incertitude se réduit alors. L'intégrité est conservée tout au long de de ce processus.
2. Ici, le véhicule essaie de détecter un mur qui devrait normalement être visible et détectable facilement. En réalité, ce mur est caché par des buissons, ce que le système ignore. La confiance dans la localisation estimée se dégrade donc vite et induit des remises en cause successives. Finalement le système détecte les murs à l'intérieur du hangar et revient à une estimation stabilisée.

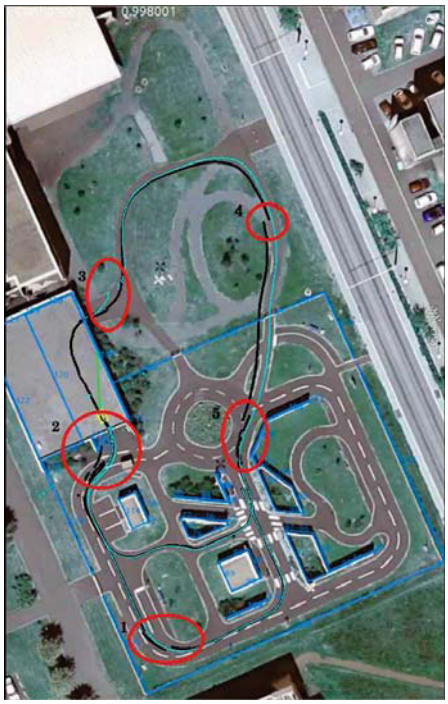


FIGURE VII.20 – Localisation dynamique, estimation de la trajectoire avec un télémètre et un capteur d'angle : évolution de l'algorithme. En noir la trajectoire estimée avec notre système, en cyan la trajectoire de référence fournie par le récepteur GPS RTK.

3. En sortant du hangar, le même phénomène que pour le cas 1 se produit. Ensuite sur toute la trajectoire parcourue dans cette zone sans mur, seul le capteur d'angle est sollicité.

4. Le véhicule essaie ici de détecter le grillage. La détection qui en résulte ne donne une information que suivant une direction donnée, la mise à jour à l'aide du filtre de Kalman décale la position estimée. Durant toute cette opération, la position estimée demeure intègre (la position réelle est à l'intérieur de l'ellipse d'incertitude centrée sur la position estimée). Le véhicule ne pourra se mettre à jour suivant d'autres directions qu'en présence d'autres amers.

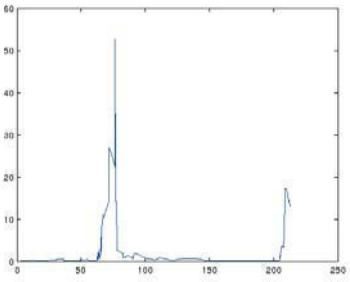
5. Nous assistons ici à la suite logique du point précédent. Le véhicule voit d'autres amers contenant des informations dans d'autres directions. Le système recale alors sa position estimée.

En dehors de la zone urbaine, il n'y a pas d'amers détectables par le télémètre laser (aucun amer cartographié), le système utilise donc dans cette zone le capteur d'angle exclusivement. Dès que le véhicule revient dans la zone urbaine, le système sélectionne le télémètre laser pour essayer de détecter d'autres amers. Tout ceci ce fait naturellement sans heuristique spécifique.

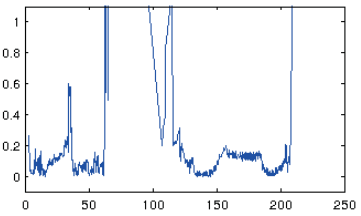
Localisation avec un télémètre et un GPS

Noter véhicule est maintenant équipé d'un télémètre laser et d'un récepteur GPS. Nous avons déjà utilisé le récepteur GPS pour simuler un capteur d'angle absolu, maintenant ce même récepteur est utilisé mais via un triplet de positionnement en abscisse et en ordonnée. Les résultats sont visibles sur la figure VII.21.

(a) Trajectoire du véhicule. En noir la trajectoire estimée avec notre système, en cyan la trajectoire de référence fournie par le récepteur GPS RTK.



(b) Erreur de localisation (en mètres) au cours du temps (en secondes)



(c) Erreur de localisation (en mètres) au cours du temps (en secondes)

FIGURE VII.21 – Localisation dynamique, estimation de la trajectoire avec un télémètre et un récepteur GPS naturel : évolution de la trajectoire. En noir, la trajectoire estimée par le système. En bleu, la trajectoire de référence fournie par le GPS RTK.

Le résultat est similaire aux données avec juste un télémètre, ce qui se comprend tout à fait car le récepteur GPS permet d'obtenir une précision de l'ordre de 5 mètres alors que le télémètre laser fournit une précision de l'ordre de quelques centimètres dans la direction privilégiée et est par conséquent le seul capteur sollicité en dehors de la première sélection.

Nous pouvons remarquer plusieurs sauts dans la trajectoire estimée. Ces sauts sont entourés d'une ellipse rouge et numérotés. Les sauts 1, 2, et 4 ont exactement les mêmes causes que les sauts 1, 2, et 5 vus dans la section VII.3.3, nous ne les détaillerons donc

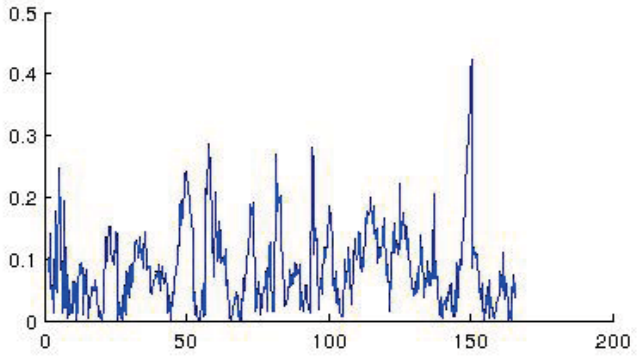
pas ici.Le saut 3 correspond à l'utilisation du capteur GPS.

Il serait tout à fait possible d'utiliser non plus le résultat fourni par le récepteur GPS mais les pseudo-distances des satellites, chaque pseudo-distance fournirait une information propre, certaines seraient fausses car potentiellement occultées, etc.

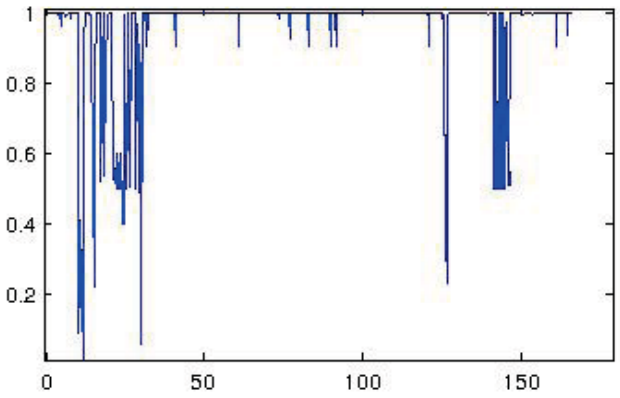
Dans la partie urbaine, les performances sont identiques à celles de l'expérimentation réalisée avec un télémètre et un capteur d'angle. Ceci est tout à fait normal puisqu'à l'intérieur de la zone urbaine seul le télémètre est sollicité par notre système car les triplets comportant le télémètre apportent une plus grande précision que le récepteur GPS.

Localisation avec quatre télémètres

Notre algorithme a été conçu pour fonctionner avec plusieurs capteurs, qui peuvent être ou non de même nature, nous le testons ici avec quatre capteurs identiques : des télémètres laser situés aux quatre coins du véhicule. Le nombre de triplets est donc potentiellement multiplié par quatre. Afin d'illustrer le résultat, nous avons parcouru une trajectoire d'environ 350 mètres représentée sur la figure VII.23.



(a) Évolution de l'erreur au cours du temps.



(b) Évolution de la confiance au cours du temps

FIGURE VII.22 – Localisation dynamique, estimation de la trajectoire avec quatre télémètres lasers : évolution de la confiance (m) et de l'erreur au cours du temps (s).

FIGURE VII.23 – Localisation dynamique, estimation de la trajectoire avec quatre télémètres lasers : évolution de l'algorithme. En noir la trajectoire estimée avec notre système, en cyan la trajectoire de référence fournie par le récepteur GPS RTK.

La précision (figure VII.22a) est moins bonne qu'avec un seul télémètre (ici on a une moyenne de l'erreur égale à 8 cm alors qu'avec un seul télémètre elle était de 6 cm soit un écart relatif de 25%), cela n'est pas du à la méthode en elle-même mais à un souci de temps de calcul. En effet, avec quatre télémètres, le nombre de triplets disponibles est multiplié par quatre environ ce qui augmente fortement le temps de calcul nécessaire. Cette augmentation du temps de calcul implique que beaucoup de données capteur ne sont pas utilisées. Ce souci pourrait être réglé par une meilleure optimisation du code qui n'est, dans l'état, pas développé de manière optimale. En particulier, le calcul du réseau bayésien pourrait être fortement accéléré avec l'utilisation d'outils dédiés à notre cas (nœuds binaires, structure fixe au cours du temps). Actuellement le temps de calcul du meilleur triplet (basé principalement sur le réseau bayésien) occupe 30 % du temps de calcul. Des travaux internes à l'équipe ont montré que cette charge de calcul pourrait être divisée par 15 environ.

Le graphe de confiance (figure VII.22b) est similaire à ceux que l'on a pu observer précédemment avec par exemple de fortes oscillations quand le véhicule essaie de détecter un mur caché par des buissons (au début et à la fin sur la figure VII.22b).

Un télémètre et initialisation incorrecte

Dans le cas présenté ici le robot est initialisé avec une erreur de 8 mètres vis-à-vis de la position réelle et une incertitude de position de 12 mètres. Nous nous sommes placés ensuite dans un cas où le processus de localisation, quand le robot ne bouge pas, converge vers une solution fausse (état-puits) puis nous avons fait avancer le robot.

Tous les points noirs sur la figure VII.24 montrent les différentes positions estimées par le robot. Très vite le robot se rend compte qu'il n'est pas au bon endroit puisqu'il n'arrive à détecter aucun des amers qu'il essaie de voir, il remet donc en cause sa position et réassocie les différentes observations qu'il a pu faire. Compte tenu de l'incertitude initiale fort grande, le système met un certain temps pour retrouver sa vraie position mais finit par y arriver (ellipse noire figure VII.24). Ensuite la localisation se

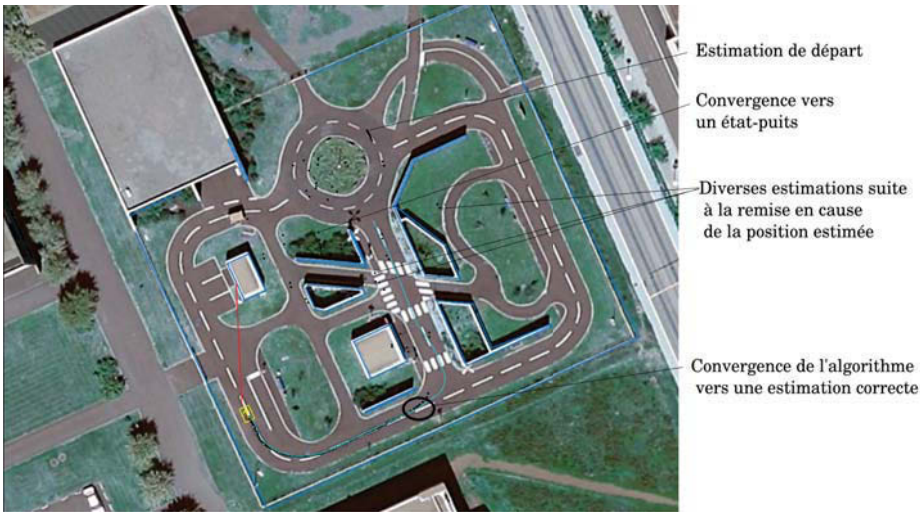


FIGURE VII.24 – Localisation dynamique, estimation de la trajectoire avec un télémètre, une estimation de départ incorrecte suite à une mauvaise association lors de la phase d'initialisation : évolution de l'algorithme. En noir, la trajectoire estimée En bleu, la trajectoire de référence fournie par le récepteur GPS RTK.

poursuit normalement. Durant tout ce temps, l'erreur varie fortement de 0 à 16 mètres. C'est là un des principaux points forts de cet algorithme : savoir détecter que sa position est très probablement fausse et être capable de revenir à une position estimée intègre.

VII.3.4 Applications

La localisation précise et fiable d'un véhicule est rarement une fin en soi et est généralement calculée pour une exploitation comme la conduite autonome de véhicules par exemple. Nous avons utilisé le résultat de la localisation pour réaliser de la conduite autonome de véhicules à travers deux applications que nous montrons ici.

Convoi de véhicules avec évitement d'obstacles

Cette application a été réalisée dans le cadre du projet SafePlatoon [1]. Trois VipaLab ont été utilisés sur lesquels l'algorithme de localisation était implémenté. Deux véhicules étaient équipés d'un télémètre laser à l'avant centre, le troisième était équipé de quatre télémètres lasers, un à chaque coin (figure VII.25a). Les véhicules sont tous les trois conduits de manière automatique.

Ces véhicules se déplacent en convoi, celui de tête étant le leader, les deux autres se contentent de suivre la trajectoire préalablement réalisée par le premier. Pour cela, les modules Wifi 802.11p, permettant la communication entre les véhicules, sont utilisés. Le leader envoie les coordonnées géoréférencées de la trajectoire qu'il parcourt, les deux autres véhicules suivent alors cette dernière.

1. Le projet SafePlatoon est un projet ANR qui aborde la problématique du fonctionnement en convoi de véhicules autonomes.

(a) Véhicules utilisés en convoi

(b) Trajectoire préalablement définie

FIGURE VII.25 – Convoi de véhicules : deux véhicules suivent un véhicule de tête. Ce dernier suit une trajectoire prédéfinie mais est capable de s'en écarter pour éviter les éventuels obstacles présents sur la route.

FIGURE VII.26 – Évolution temporelle du convoi de véhicules avec évitement d'obstacles (à lire de gauche à droite et de haut en bas).

Le chemin suivi par le leader est prédéfini (figure VII.25b) lors d'une première phase d'apprentissage mais est susceptible de comporter des obstacles. Le leader détecte ces obstacles et est alors capable de changer dynamiquement sa trajectoire afin d'éviter l'obstacle puis de revenir sur la trajectoire prédéfinie [150]. Les deux autres véhicules se contentent de suivre la trajectoire du leader. Nous pouvons voir l'évolution des véhicules sur la figure VII.26.

Récupération de trajectoire, changement de leader

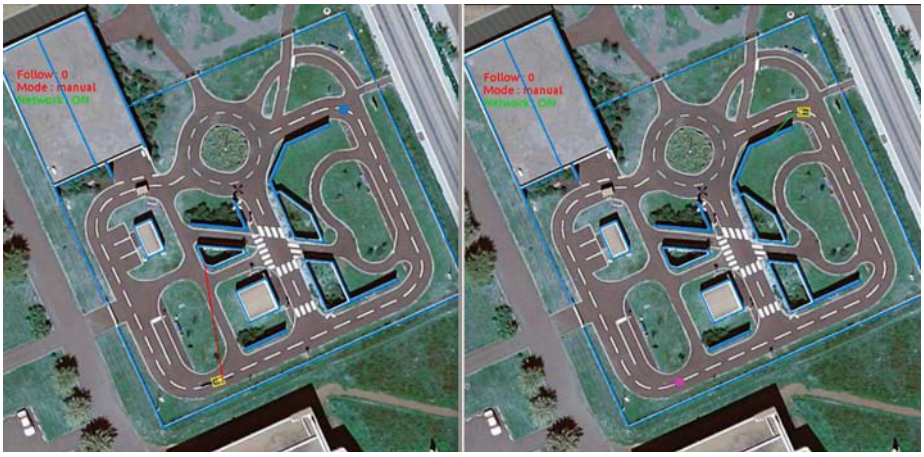


FIGURE VII.27 – Navigation automatique avec récupération de trajectoire : situation de départ. À gauche, le véhicule vipalab3 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab5. À droite, le véhicule vipalab5 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab3.

Dans cet exemple, nous avons deux véhicules (nommés vipalab3 et vipalab5) qui communiquent entre eux grâce aux modules Wifi-P. Cette fois les véhicules ne sont pas exclusivement guidés de manière automatique, tout au long du déroulement de l'opération ils peuvent être guidés de manière manuelle ou automatique. Au départ, les deux véhicules partent d'endroits différents de la carte et sont commandés manuellement (figure VII.27). La localisation de chacun des deux véhicules est estimée, enregistrée et transférée en temps-réel à l'autre véhicule.

Quand un des deux véhicules rencontre une trajectoire préalablement parcourue par un autre véhicule, c'est-à-dire dans notre cas que sa position estimée est à moins de 50 centimètres de la trajectoire précédemment parcourue par l'autre véhicule, il choisit alors de la suivre de manière automatique. C'est ce que nous pouvons voir sur la figure VII.28, le vipalab3 suit la trajectoire du vipalab5 de manière automatique, ce dernier continue d'être guidé automatiquement.

À tout moment, le conducteur peut intervenir pour conduire lui-même le véhicule. Avec cette façon de faire, il n'y a plus de notion de leader ou plus exactement le leader peut changer au cours de l'opération. Sur la figure VII.29, le vipalab5 a rejoint la trajectoire du vipalab3 et suit alors cette dernière. Nous avons alors le vipalab3 qui suit une trajectoire précédente du vipalab5 et inversement.

Une vidéo est disponible à l'adresse suivante : `https://youtu.be/W0OpbxEJljg`.

Bilan des applications

Les deux applications de conduite automatique montrées illustrent bien l'utilisation qu'il peut être fait de l'algorithme de localisation. Elles démontrent en particulier que ce dernier est tout à fait opérationnel et fonctionne en temps-réel. Elles montrent aussi la généricité de cet algorithme puisqu'il implémenté sur des véhicules possédant des

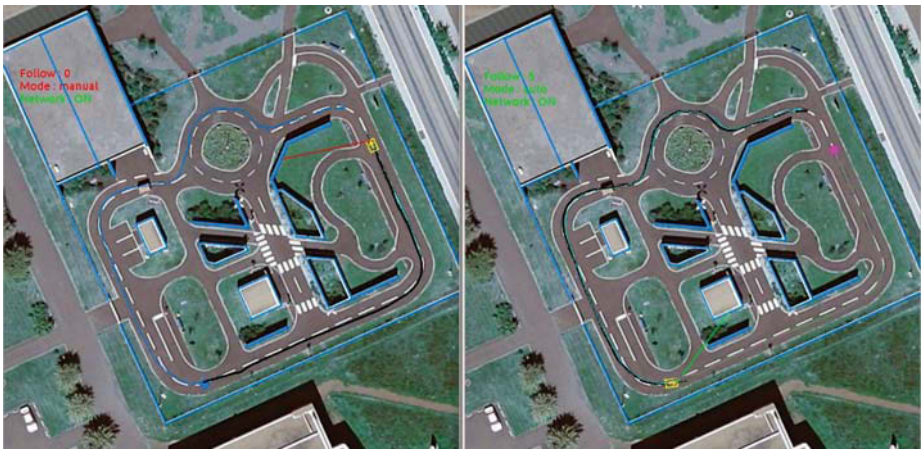


FIGURE VII.28 – Navigation automatique avec récupération de trajectoire : suivi de la trajectoire du vipalab5 par le vipalab3. À gauche, le véhicule vipalab3 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab5, ce dernier se trouve sur la trajectoire du vipalab3 et la suit. À droite, le véhicule vipalab5 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab3

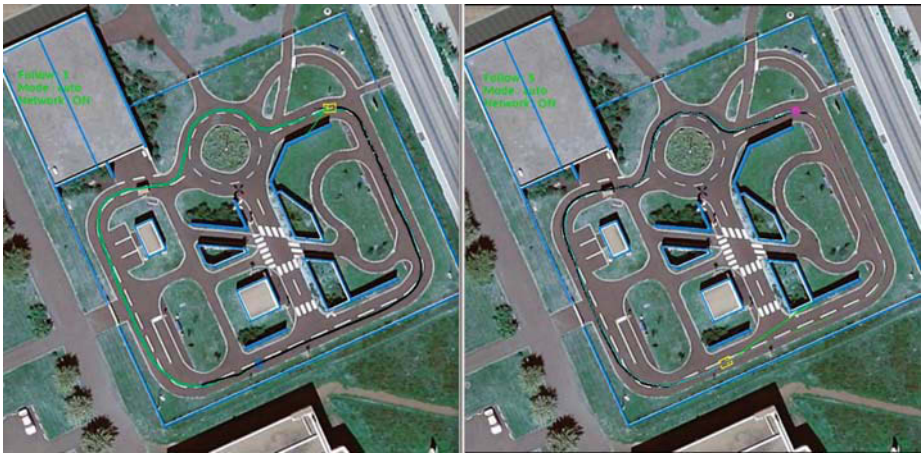


FIGURE VII.29 – Navigation automatique avec récupération de trajectoire : les deux véhicules suivent la trajectoire précédente de l'autre. À gauche, le véhicule vipalab3 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab5. À droite, le véhicule vipalab5 (en jaune) avec sa trajectoire estimée (en noir) est visible ainsi qu'une pastille bleue montrant le vipalab3

capteurs différents (un véhicule possède quatre lidars tandis que les autres n'en possèdent qu'un) et situés à des endroits différents (les quatre lidars sont montés aux quatre coins du véhicule tandis que les véhicules avec un seul lidar l'ont au milieu du véhicule vers l'avant). Le fait que la position estimée soit géoréférencée simplifie grandement les interactions entre les véhicules et élargit le champ d'applications.

VII.4 Conclusion

Dans cette partie, nous avons présenté en premier lieu comment notre algorithme a été implémenté et quelles étaient les ressources disponibles. Dans un second temps nous avons présenté les différents résultats et les applications réalisées. Nous avons vu que l'algorithme développé permettait à la fois de réaliser une localisation statique (permettant d'illustrer son comportement en situation de robot kidnappé) et une localisation dynamique. Pour l'algorithme, il n'y a pas de différence entre ces deux étapes ce qui est un point fort de l'approche puisqu'il n'y a pas besoin d'une étape spécifique pour l'initialisation. Il y a cependant un bémol pour l'initialisation concernant l'orientation du véhicule, si l'orientation initiale est trop éloignée de la réalité (plus de 70° environ), l'algorithme ne peut converger notamment à cause de la non-linéarité intrinsèque du filtre de Kalman. La confiance est toujours faible lorsque la position estimée n'est pas intègre, ce qui est permet une remise en cause rapide. Un autre point fort est le côté générique du système. Qu'il y ait un ou plusieurs capteurs (et leurs détecteurs associés), que ces capteurs soient identiques ou non, le système réagit toujours de manière à prendre en compte le meilleur triplet perceptif à chaque instant, c'est-à-dire le meilleur capteur, le meilleur détecteur et le meilleur amer. Enfin des applications de navigation ont été présentées. Ces applications ont permis de montrer que la méthode est applicable en temps-réel et qu'elle fournit une localisation suffisamment précise et fiable pour réaliser de la navigation automatique seul ou en convoi. De plus, l'utilisation d'une localisation géoréférencée facilite les communications entre les véhicules et élargit le champ des possibilités.

CHAPITRE VIII

Conclusion et Perspectives

Sommaire

VIII.1 Conclusion

Dans ce manuscrit, nous sous sommes intéressés à la localisation de véhicules à partir d'une carte. Nous avons vu au chapitre 3 qu'il nous semblait pertinent de s'orienter sur une approche multi-sensorielle top-down afin d'exploiter au mieux les informations disponibles. Toute l'application développée est ici dirigée par notre double objectif : assurer une bonne précision et une bonne confiance dans la localisation fournie par le système. La mesure des performances de notre système ne s'effectue donc pas seulement à travers l'estimation de sa précision mais également sur sa capacité à fournir le degré de confiance de l'estimation, en d'autres termes sa capacité d'auto-critique.

Dans un premier temps, nous avons donc mis en place un critère de sélection (chapitre IV) qui permet à chaque instant de sélectionner parmi toutes les informations disponibles celle qui semble la plus pertinente vis-à-vis de l'objectif et du contexte. Ce critère, semblable en plusieurs points à celui d'un être humain dans la même situation, permet ainsi d'améliorer les performances du système de détection. Cette stratégie de sélection permet également d'avoir un système focalisant améliorant ainsi substantiellement les performances de la perception et diminuant les risques de fausse association.

Afin de bien prendre en compte les différents événements présents et leurs interactions, nous avons mis en place un réseau bayésien dynamique discret (chapitre V). Ce réseau rend la compréhension des phénomènes en action plus aisée et fournit un cadre mathématique rigoureux. En outre, l'évolution d'un tel réseau est aisée à réaliser et permettra dans l'avenir de prendre facilement en compte d'autres événements.

De par le système d'inférence du réseau bayésien, il est possible de déterminer les causes probables d'une mauvaise association. En partant de ce principe, un système de détection d'erreurs et de réparation des erreurs a été mis en place (chapitre VI). Ainsi même quand la position estimée est fausse, le système est capable de le détecter et d'y remédier. Cette technique rend possible la résolution de scénarios critiques notamment dans le cas du robot kidnappé (chapitre VII section VII.3.2). Elle permet d'avoir un système fonctionnel y compris quand l'incertitude initiale est très grande et par là même permet d'éviter une phase d'initialisation spécifique.

Cette méthode de localisation a été appliquée dans des cas réels avec des véhicules possédant différents types de capteurs (chapitre VII). Les résultats ont permis de montrer l'efficacité de cette méthode. Des applications pratiques s'appuyant sur le module de localisation ont pu être mises en place comme de la conduite autonome en convoi.

Ce travail de recherche a fait l'objet de plusieurs publications nationales et internationales (voir page 165). Le principe de cette méthode top-down a également été utilisé dans un autre domaine en interne, à savoir en reconnaissance de formes sur une image [5].

VIII.2 Perspectives

Les résultats des travaux présentés sont prometteurs mais ils peuvent toutefois être encore améliorés. Le but de cette partie est de présenter quelques pistes d'amélioration qui nous semblent pertinentes, c'est-à-dire apportant une plus value potentiellement élevée et réalisables.

VIII.2.1 Prise en compte du temps dans le critère de sélection

Dans le cadre de cette thèse peu de détecteurs ont été utilisés et tous étaient très rapides. Cependant avec l'ajout de nouveaux capteurs, ou de nouveaux types d'amers, il est possible d'imaginer que les détecteurs peuvent devenir assez variés et avoir un temps d'exécution non négligeable. La prise en compte de ce dernier directement dans le critère est donc à envisager. Supposons par exemple deux détecteurs qui ont comme but de détecter le même type d'amer. Le premier détecteur possède les caractéristiques suivantes : très forte précision, très forte probabilité de détecter effectivement l'amer s'il est visible, très forte probabilité de ne rien détecter s'il n'est pas visible mais en revanche un temps de calcul exponentiel suivant la taille de la zone de recherche. Le second détecteur possède quant à lui des caractéristiques moyennes mais un temps d'exécution beaucoup plus rapide. Si l'incertitude initiale est très grande, le premier détecteur risque de prendre beaucoup trop de temps, il est donc plus judicieux ici de choisir le deuxième détecteur dans un premier temps et seulement ensuite le second. Avec le critère défini actuellement, le premier détecteur aurait été choisi puisque le temps n'est pas pris en compte. Cette prise en compte pourrait se faire directement dans le critère en s'inspirant par exemple de ce qui a été fait dans [5].

VIII.2.2 Objectif de précision dynamique

L'objectif de précision est défini suivant l'application et reste ensuite statique. Il serait intéressant de le faire évoluer en fonction des besoins. Par exemple, pour une conduite sur autoroute une certaine précision latérale est requise pour rester dans la voie mais dans le sens longitudinal (sens du véhicule), elle peut être plus faible. Au niveau d'un stop en revanche, il sera nécessaire d'avoir une forte précision dans toutes les directions. Un objectif dynamique pourrait ainsi être mis en place et varier au cours du temps suivant l'environnement. Cette façon de faire peut avoir plusieurs avantages parmi lesquels :

- la sélection de triplets de façon plus pertinente pour l'instant présent ;
- une utilisation plus large du système.

Le premier cas se comprend aisément. Sur l'autoroute, le véhicule aura tendance à détecter les lignes blanches latérales au sol en priorité tandis qu'à l'approche d'un stop, il va par exemple chercher prioritairement à détecter le panneau de signalisation. Dans le deuxième cas, nous prenons ici en compte le fait que le système final est opérant si et seulement si la brique de localisation assure une localisation suffisamment précise et fiable. Le terme suffisamment dépend beaucoup du contexte. Ce qui peut être suffisant par exemple pour de la conduite sur route dégagée peut ne pas l'être à l'approche d'un stop. Avoir un niveau d'exigence différent permettra d'utiliser ce système dans plus de situations.

VIII.2.3 Détection de capteurs défectueux

Dans [36], les auteurs montrent un robot capable d'adapter son comportement si une de ses pattes est abîmée ou cassée. De manière similaire, il serait intéressant de développer un système de détection des capteurs défectueux et de s'adapter en conséquence. En effet, les tests réalisés sur les véhicules sont toujours effectués sur des véhicules contrôlés, équipés spécialement pour la recherche. Cependant, l'utilisation d'un tel algorithme dans un contexte industriel implique que le véhicule soit capable de réagir face à un capteur devenu défectueux, c'est-à-dire a minima d'identifier quel capteur est défectueux. Savoir qu'un capteur est défectueux permettra au système d'une

part de ne plus le solliciter et d'autre part de le signaler. Ceci pourrait être réalisé grâce au réseau bayésien. Imaginons par exemple qu'un véhicule n'arrive pas à détecter un amer, il pourra mettre ceci sur le compte de l'existence de l'amer ainsi que du capteur concerné, si maintenant il arrive plus tard à détecter ce même amer avec un autre capteur, alors la probabilité que l'échec de la détection de l'amer soit dû à l'absence de l'amer diminue drastiquement tandis qu'a contrario la probabilité que le capteur soit défectueux augmente.

VIII.2.4 Correction et enrichissement de la carte

La carte utilisée ici est une carte très précise qui a permis de tester et valider le principe de la méthode. Toutefois, une telle carte est rarement disponible en tout lieu et en tout temps. Il serait intéressant d'avoir une méthode capable d'utiliser une carte déjà fournie type Open Street Map. Ce genre d'amélioration permettrait d'avoir un véhicule capable de se localiser à peu près n'importe où. Un des principaux inconvénients de ce type de carte est la précision des données ainsi que la certitude de ces données. Certaines données peuvent être très précises et fiables tandis que d'autres ne seront fournies que de manière très approximatives. Il peut arriver que la position du véhicule soit plus précise que celle des données fournies par la carte. Dans ce cas, au lieu d'utiliser les amers pour localiser le véhicule le processus inverse pourrait être appliqué : utiliser la position du véhicule pour localiser précisément les amers. Finalement, le processus de localisation se transformerait en un processus de diminution de l'imprécision globale, l'objectif principal étant la localisation précise et fiable du véhicule et l'objectif secondaire étant la localisation précise et fiable des amers.

Le problème de la précision des données a été abordé mais il reste celui de l'existence des données cartographiées. Cela prend deux aspects différents :

- un amer cartographié est en réalité absent ;
- un amer en réalité présent n'est pas cartographié.

Dans le premier cas, il peut arriver qu'un amer ayant été présent au moment de la cartographie ne soit plus présent en réalité ou bien qu'il ait été cartographié par erreur. Si le système n'arrive jamais à détecter cet amer, quel que soit l'angle sous lequel il essaie de le voir, et quel que soit le capteur alors il pourra en déduire qu'il n'existe pas et l'éliminer de la carte. Ceci pourrait être fait en attribuant à chaque amer une probabilité d'existence qui va être mise à jour lors des essais de détections de cet amer. Lors de détections échouées, il sera très important pour le système de bien diagnostiquer l'origine de ce dysfonctionnement qui peut provenir de plusieurs choses comme notamment la non-intégrité de la position estimée du robot, de l'absence de l'amer, de l'occultation de l'amer par un autre élément non-cartographié ou encore d'un capteur défectueux. Grâce au réseau bayésien mis en place, il devrait être possible de prendre en compte ces différents éléments et essayer d'identifier au mieux la source du problème.

Dans le second cas, il s'agit d'enrichir la carte. Ce processus pourrait être déclenché quand la position estimée du véhicule est suffisamment précise et permet donc de cartogarphier de manière précise l'environnement. Une autre source de déclenchement d'un tel processus pourrait être l'échec répété de détection d'un amer suivant un certain angle. Si à chaque fois que l'algorithme essaie de détecter un amer sous un certain point de vue, cet essai échoue alors que sous un autre point de vue il réussit c'est certainement qu'il y a un objet occultant entre le capteur et l'amer recherché. Il pourrait donc être intéressant de garder en mémoire la présence de cet objet occultant et donc de mieux calculer la probabilité de détecter l'amer occulté. Le processus se rapproche

alors des méthodes de type SLAM. On aboutirait alors à un processus global à la fois Top-Down et Bottom-Up, se rapprochant de plus en plus du comportement humain.

VIII.2.5 Modélisation 3D de l'environnement

Dans tout le processus de cette thèse, le robot a été considéré comme évoluant dans un monde en deux dimensions. Cette approximation permet une estimation de la localisation correcte dans la plupart des cas, cependant cette estimation pourrait être affinée par la prise en compte du relief. En outre, une modélisation du monde en 3 dimensions permettrait d'intégrer la caméra plus facilement. En effet dans les capteurs utilisés et présentés, la caméra n'est pas présente. Des tests ont été réalisés avec ce capteur notamment pour la détection de lignes blanches de marquage au sol, qui sont également accessibles via notre Système d'Information Géographique. Du fait de la non prise en compte du relief dans notre modèle de l'environnement, des problèmes de non-linéarité surviennent dans les zones pentues, présentes sur le site d'expérimentation PAVIN. La prise en compte de l'environnement en 3D devrait, à notre avis, pallier ce problème.

ANNEXE A

Le filtre de Kalman

Le filtre de Kalman est un estimateur très populaire [71] qui, à partir d'observations successives entachées d'un bruit gaussien, cherche à estimer les paramètres d'un système. Ces paramètres peuvent être la position d'un véhicule, d'amers par exemple. L'ensemble des paramètres du système est appelé état et est noté X, la matrice de covariance associée est notée P. Le filtre de Kalman est simple à mettre en place, optimal pour les systèmes linéaires et peu coûteux. Ce filtre est un estimateur récursif, l'état à l'étape $k+1$ dépend uniquement de l'état à l'instant k et des observations à l'instant $k+1$. Deux grandes étapes sont à distinguer : la prédiction et la mise à jour.

A.1 Le filtre de Kalman linéaire

A.1.1 Phase de prédiction

Soit un système linéaire défini comme suit : $X_{k+1|k} = \mathbf{F}_{k+1} X_{k|k} + \mathbf{B}_{k+1} u_{k+1} + v_{k+1}$
Alors la prédiction est réalisée en utilisant les équations suivantes :

$$\begin{array}{ll} X_{k+1|k} & = \mathbf{F}_{k+1} X_{k|k} + \mathbf{B}_{k+1} u_{k+1} + v_{k+1} \\ \mathbf{P}_{k+1|k} & = \mathbf{F}_{k+1} \mathbf{P}_{k|k} \mathbf{F}_{k+1}^T + \mathbf{Q}_{k+1} \end{array} \tag{A.1}$$

où :
- $\mathbf{F}_{k+1}$ est la forme matricielle du modèle linéaire d'évolution de l'état ;
- u_{k+1} les commandes reçues et effectuées par le robot entre l'instant k et $k+1$;
- $\mathbf{B}_{k+1}$ la forme matricielle reliant les commandes reçues à l'état ;
- $\mathbf{Q}_{k+1}$ est la matrice de covariance du bruit v_{k+1} d'évolution du processus.

Dans notre cas, nous utiliserons les données d'odométrie pour prédire la position du véhicule. En particulier, nous utiliserons les déplacements relatifs des roues ainsi que l'angle de braquage.

A.1.2 Phase de mise à jour

Lors de la phase de mise à jour le système va prendre en compte les observations réalisées au cours du temps. On note z_k l'observation défini de la manière suivante :

$z_{k+1} = \mathbf{H}_{k+1} X_{k+1} + w_{k+1}$. Le système d'équation utilisé est alors le suivant :

$$\begin{array}{ll} \mathbf{G}_{k+1} & = \mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T(\mathbf{H}_{k+1}\mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T + \mathbf{R}_{k+1})^{-1} \\ X_{k+1|k+1} & = X_{k+1|k} + \mathbf{G}_{k+1}(\mathbf{z}_{k+1} - \mathbf{H}_{k+1}X_{k+1|k}) \\ \mathbf{P}_{k+1|k+1} & = \mathbf{P}_{k+1|k} - \mathbf{G}_{k+1}\mathbf{H}_{k+1}\mathbf{P}_{k+1|k} \end{array} \tag{A.2}$$

où :
- z_{k+1} est la variable regroupant les observations faites à l'instant $k+1$;
- $\mathbf{H}_{k+1}$ est la forme matricielle du modèle d'observation reliant z_{k+1} à l'état ;
- $\mathbf{R}_{k+1}$ est la matrice de covariance du bruit w_{k+1} d'observation ;
- $\mathbf{G}_{k+1}$ désigne le gain de Kalman.

A.2 Filtre de Kalman étendu

Le filtre de Kalman est conçu pour fonctionner dans le cas de systèmes linéaires. Dans le cas de systèmes non-linéaires ce qui est souvent le cas en robotique, une extension du filtre de Kalman est utilisée, le filtre de Kalman étendu (EKF en anglais pour *Extended Kalman Filter*) [70]. Il agit de manière très similaire au filtre de Kalman classique mais en ajoutant une étape de linéarisation.

A.2.1 Phase de prédiction

Soit un système défini comme suit : $X_{k+1|k} = f(X_{k|k}, u_{k+1}, v_k)$. Les équations du filtre de Kalman sont alors les suivantes :

$$\begin{array}{ll} X_{k+1|k} & = f(X_{k|k}, u_{k+1}) \\ \mathbf{P}_{k+1|k} & = \mathbf{F}_{k+1}\mathbf{P}_{k|k}\mathbf{F}_{k+1}^T + \mathbf{Q}_{k+1} \end{array} \tag{A.3}$$

où $\mathbf{F}_{k+1}$ est la jacobienne de la fonction d'évolution $f(X_{k|k}, u_{k+1})$ qui tient compte de l'état précédent et des entrées u_{k+1} du système. Les autres paramètres sont les mêmes que dans le cas linéaire.

Dans notre cas, nous utiliserons les données d'odométrie pour prédire la position du véhicule. En particulier, nous utiliserons les déplacements relatifs des roues ainsi que l'angle de braquage.

A.2.2 Phase de mise à jour

Lors de la phase de mise à jour le système va prendre en compte les observations réalisées au cours du temps. La fonction d'observation est définie de la manière suivante : $z_{k+1} = h(X_{k+1}, w_{k+1})$. Les équations utilisées sont alors :

$$\begin{array}{ll} \mathbf{G}_{k+1} & = \mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T(\mathbf{H}_{k+1}\mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^T + \mathbf{R}_{k+1})^{-1} \\ X_{k+1|k+1} & = X_{k+1|k} + \mathbf{G}_{k+1}(\mathbf{z}_{k+1} - \mathbf{h}(X_{k+1|k})) \\ \mathbf{P}_{k+1|k+1} & = \mathbf{P}_{k+1|k} - \mathbf{G}_{k+1}\mathbf{H}_{k+1}\mathbf{P}_{k+1|k} \end{array} \tag{A.4}$$

où $\mathbf{H}_{k+1}$ est la jacobienne de la fonction d'observation $\mathbf{h}(X_{k+1|k})$. Les autres paramètres sont les mêmes que dans le cas linéaire.

A.2.3 Utilisation du filtre de Kalman étendu

Ce modèle peut entraîner de graves soucis de divergence si le point estimé est trop éloigné du point réel. D'autres approches existent pour ce genre de problématique comme le filtre de Kalman sans parfum (UKF en anglais pour *Unscented Kalman Filter*) [68]. L'idée maîtresse du filtre UKF est d'éviter le calcul des jacobiennes de l'EKF par l'introduction d'une phase d'échantillonnage amenant à l'introduction de particules, chaque particule étant estimée indépendamment. Généralement plus précis que l'EKF, l'UKF nécessite un temps de calcul plus important. De plus il a été montré dans [53] qu'utiliser un EKF convenait dans la plupart des situations.

ANNEXE B

Propagations dans les réseaux bayésiens : algorithme JLO

Il existe plusieurs algorithmes permettant de calculer les différentes probabilités des événements modélisés par un réseau bayésien. Les deux plus connus sont les algorithmes de Pearl et JLO (ce dernier est aussi appelé algorithme de Jensen). L'algorithme de Pearl est le premier historiquement développé et travaille directement sur le graphe. L'algorithme de Jensen en diffère principalement par le fait qu'il travaille sur un graphe différent obtenu à partir du premier et se prêtant mieux à la propagation d'informations. C'est ce dernier qui est mis en œuvre dans la bibliothèque dlib utilisée dans le cadre de cette thèse. C'est pourquoi c'est cet algorithme qui est décrit ici. Dans un premier temps, les définitions nécessaires à la compréhension de l'algorithme sont données, les notions déjà vues dans la partie III.4.3 ne sont pas redonnées. Dans un deuxième temps, le déroulement de l'algorithme est présenté et illustré au fur et à mesure avec un réseau simple. La description de cet algorithme est fortement inspiré des descriptions réalisées dans [135] et [106]. Les fondements théoriques de l'algorithme ne sont pas abordés ici.

B.1 Notions sur la théorie des graphes nécessaires pour l'algorithme JLO

Définition B.1 (Moralisation et famille)
La moralisation d'un graphe orienté consiste à relier chaque paire de nœuds ayant un nœud fils commun. Le graphe obtenu est un graphe non orienté appelé graphe moral et dont les arcs sont transformés en arêtes. L'ensemble constitué du nœud et de ses parents est appelé famille.

Définition B.2 (Clique)
Une clique dans un graphe non orienté G est un sous graphe de G complet et maximal. Il est complet si chaque paire de nœuds distincts est reliée par une arête.

Il est maximal s'il n'est pas contenu dans un plus grand sous graphe complet.

Définition B.3 (Arbre)
Un graphe $G = (V,E)$ est un arbre si et seulement s'il est connexe et sans cycle.

Définition B.4 (Graphe de jonction)
Soit $G = (V,E)$ un graphe non orienté, soit $C = \{C_1, C_2, \ldots, C_n\}$ l'ensemble des cliques de G tel que $V = C_1 \cup C_2 cup \ldots \cup C_n$. On nomme graphe de jonction le graphe (C, E_c) vérifiant : $(C_1, C_2) \in E_c \Longleftrightarrow C_1 \cap C_2 \neq \emptyset$. On nomme $S_{12} = C_1 \cap C_2$ le séparateur C_1 et C_2. Un graphe de jonction est dit minimal s'il ne possède aucun

Définition B.5 (Arbre de jonction)
Un graphe de jonction qui est également un arbre est appelée arbre de jonction.

B.2 Description de l'algorithme JLO et illustration sur un exemple simple

B.2.1 Introduction

L'algorithme JLO s'applique à des réseaux ne comprenant que des variables à valeurs discrètes. Il existe cependant des extensions pour des distributions gaussiennes et des mixtures de gaussiennes [41].

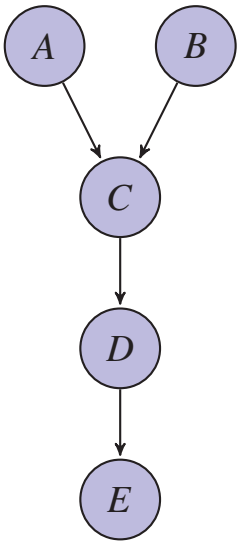
Cet algorithme comporte deux grandes parties :

- la phase de transformation du graphe initial : le graphe initial fait l'objet de plusieurs algorithmes visant à le transformer en arbre de jonction dont les nœuds sont des regroupements de nœuds du graphe initial ;
- la phase de propagation : il s'agit de la phase de calcul probabiliste durant laquelle les informations concernant une ou plusieurs variables sont propagées à l'ensemble du réseau de manière à mettre à jour l'ensemble des distributions de probabilités du réseau.

Afin de comprendre le déroulement de l'approche, nous illustrons chaque étape sur un réseau Bayésien simple dont l'architecture et les tables de probabilités conditionnelles sont donnés sur la figure B.1. Ce réseau Bayésien est composé de nœuds discrets binaires.

B.2.2 Transformation du graphe initial

La phase de transformation du graphe initial se fait en trois étapes :



P(A) = 0.8
P(B) = 0.1

A	B	$P(C)$
0	0	0.2
0	1	0.6
1	0	0.7
1	1	0

C	$P(D)$
0	0.2
1	0.6

D	$P(E)$
0	0.6
1	0.9

FIGURE B.1 – Réseau Bayésien utilisé pour illustrer le processus de l'algorithme JLO et tables de probabilités associées

– la moralisation du graphe ;
– la triangulation du graphe et l'extraction des cliques ;
– construction de l'arbre de jonction.

Moralisation du graphe

La première étape de l'algorithme consiste à moraliser le graphe initial, c'est-à-dire à relier deux à deux les parents de chaque variable par un arc non-dirigé. Le graphe obtenu n'est pas un graphe orienté. Le réseau Bayésien G est alors transformé en un graphe montré sur la figure B.2.

Triangulation et extraction des cliques

La deuxième étape consiste à trianguler le graphe moral obtenu et à en extraire des cliques de nœuds. Il existe plusieurs algorithmes de triangulation dans la littérature [60, 77, 139]. Dans l'exemple qui nous concerne, il n'existe pas de cycle de longueur 4 ou plus, le graphe est déjà triangulé. Si ce n'était pas le cas, il faudrait rajouter des arcs dans le graphe pour respecter cette propriété.

Il faut maintenant identifier les cliques présentes dans l'arbre, c'est-à-dire chaque sous-graphe complet et maximal de l'arbre triangulé. Les cliques identifiées sont décrites dans la table B.1.

Construction de l'arbre de jonction

La quatrième étape consiste alors à relier les cliques entre elles (figure B.3). Le graphe obtenu est appelé graphe de jonction. Il faut ensuite le transformer en un arbre

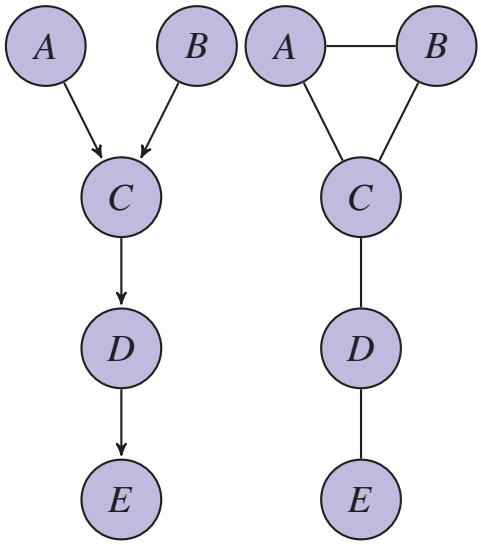


FIGURE B.2 – Transformation du graphe initial (à gauche) en un graphe moral (à droite)

	Cliques
C1	{A, B, C}
C2	{C, D}
C3	{D, E}

TABLE B.1 – Identification des cliques du graphe moral de la figure B.2

de jonction. Pour notre exemple, cela est immédiat, si ce n'était pas le cas il faudrait enlever des arcs redondants en appliquant l'algorithme suivant :

Données : un graphe G triangulé Résultats : Arbre de jonction de m cliques associé à G
Initialisation : pour une numérotation des cliques de i=1..m i=m
Organisation des cliques sous forme de chaîne de cliques tant que $(1 \leq i)$ faire Pour chaque clique $C_i \in C$ choisir une clique C_k parmi $\{C_1, \ldots, C_{i-1}\}$ telle que le cardinal de $|C_i \cap C_k|$ soit maximal. Ajouter le lien entre C_i et C_k $i \longleftarrow i-1$ Fin Tant que
Définir l'ensemble des séparateurs comme suit : $S_i = C_i \cap C_{i+1}$

A partir de là, il n'y a plus de modifications à réaliser sur le graphe. L'arbre de jonction obtenu est directement utilisé pour réaliser les calculs requis.

B.2.3 Propagation dans l'arbre de jonction

La propagation dans l'arbre de jonction repose sur la notion de potentiels et sur la factorisation en potentiels de cliques et séparateurs.

Initialisation de l'arbre de jonction

Chaque événement est associé à une et une seule clique, ici :

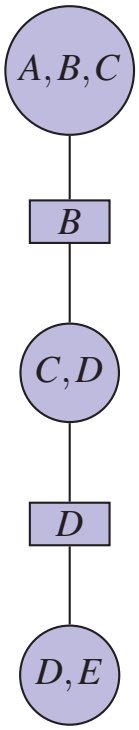


FIGURE B.3 – Arbre de jonction créé à partir du graphe moral précédent. A l'intérieur des ronds sont marquées les cliques et à l'intérieur des carrés les séparateurs

- $C_1 = \{A,B,C\}$ à cette clique on affecte les variables A, B et C ;
- $C_2 = \{C,D\}$, la variable C étant déjà affectée à C_1, seule la variable D est affectée à C_2 ;
- $C_3 = \{D,E\}$ à cette clique on affecte la variable E.

Ceci étant fait, il faut maintenant initialiser les potentiels (notés ψ) de chaque clique et séparateur. Pour cela, seuls les nœuds précédemment associés à chaque clique sont pris en compte :

- $C_1 = \{A,B,C\}$, $\psi(C_1) = P(C|A,B)P(A)P(B)$;
- $C_2 = \{C,D\}$, $\psi(C_2) = P(D|C)$;
- $C_3 = \{D,E\}$, $\psi(C_1) = P(E|D)$;

Le tableau B.2 donne les potentiels des différentes cliques.

Les potentiels des différents séparateurs sont initialisés à 1 :

- $S_1 = C$, $\psi(S_1) = 1$;
- $S_2 = D$, $\psi(S_2) = 1$;

Propagation des flux dans l'arbre

Maintenant que l'arbre de jonction est initialisé, l'étape suivante consiste à faire circuler l'information entre les différentes cliques de façon à obtenir un état d'équilibre. Quand l'information est propagée entre deux cliques, on parle de passage de flux. Après un passage de flux, les différents potentiels des cliques et des séparateurs sont mis à jour d'après la formule vue dans la partie précédente. Le flux d'information est illustré sur la figure B.4.

A	B	C	$\psi(C_1)$
0	0	0	0.144
0	0	1	0.036
0	1	0	0.006
0	1	1	0.014
1	0	0	0.256
1	0	1	0.432
1	1	0	0.08
1	1	1	0

C	D	$\psi(C_2)$
0	0	0.8
0	1	0.2
1	0	0.4
1	1	0.6

D	E	$\psi(C_3)$
0	0	0.4
0	1	0.6
1	0	0.1
1	1	0.9

TABLE B.2 – Potentiels initiaux des cliques

Table des potentiels initiale

A	B	C	$\psi(C_1)$
0	0	0	0.144
0	0	1	0.036
0	1	0	0.006
0	1	1	0.014
1	0	0	0.256
1	0	1	0.432
1	1	0	0.08
1	1	1	0

C	$\psi(S_1)$
0	1
1	1

C	D	$\psi(C_2)$
0	0	0.8
0	1	0.2
1	0	0.4
1	1	0.6

D	$\psi(S_2)$
0	1
1	1

D	E	$\psi(C_3)$
0	0	0.4
0	1	0.6
1	0	0.1
1	1	0.9

Propagation du flux de la gauche vers la droite

A	B	C	$\psi(C_1)$
0	0	0	0.144
0	0	1	0.036
0	1	0	0.008
0	1	1	0.012
1	0	0	0.216
1	0	1	0.504
1	1	0	0.08
1	1	1	0

C	$\psi(S_1)$
0	0.518
1	0.482

C	D	$\psi(C_2)$
0	0	0.4144
0	1	0.1036
1	0	0.1928
1	1	0.2892

D	$\psi(S_2)$
0	0.6072
1	0.3928

D	E	$\psi(C_3)$
0	0	0.24288
0	1	0.36432
1	0	0.03928
1	1	0.35352

FIGURE B.4 – Passage de flux dans le réseau Bayésien R

Nous avons ici un réseau Bayésien très simple et un seul passage de flux permet d'arriver à l'équilibre (n passage de flux dans l'autre sens ne changera aucune valeur de potentiel). Il est maintenant possible de calculer chaque probabilité grâce à la clique qui lui appartient. Par exemple, $P(E=1) = \sum_{E=1} \psi(C_2) = 0.36432 + 0.35352 = 0.71784$.

Introduction d'une observation

Nous allons maintenant insérer une évidence dans le réseau Bayésien. Cette évidence est choisie pour l'exemple comme étant $D=1$. D est associé à la clique C_2, c'est donc cette dernière qui est affectée. Les différentes étapes sont données sur la figure B.5.

Une fois l'évidence intégrée et la propagation de flux effectuée, il est possible de calculer n'importe quelle probabilité sachant que $D=1$. Par exemple, calculons

Table des potentiels modifiée

A	B	C	$\psi(C_1)$
0	0	0	0.144
0	0	1	0.036
0	1	0	0.008
0	1	1	0.012
1	0	0	0.216
1	0	1	0.504
1	1	0	0.08
1	1	1	0

C	$\psi(S_1)$
0	0.518
1	0.482

C	D	$\psi(C_2)$
0	0	0
0	1	0.1036
1	0	0
1	1	0.2892

D	$\psi(S_2)$
0	0.6072
1	0.3928

D	E	$\psi(C_3)$
0	0	0.24288
0	1	0.36432
1	0	0.03928
1	1	0.35352

Propagation du flux de la gauche vers la droite

A	B	C	$\psi(C_1)$
0	0	0	0.144
0	0	1	0.036
0	1	0	0.008
0	1	1	0.012
1	0	0	0.216
1	0	1	0.504
1	1	0	0.08
1	1	1	0

C	$\psi(S_1)$
0	0.518
1	0.482

C	D	$\psi(C_2)$
0	0	0
0	1	0.1036
1	0	0
1	1	0.2892

D	$\psi(S_2)$
0	0
1	0.3928

D	E	$\psi(C_3)$
0	0	0
0	1	0
1	0	0.03928
1	1	0.35352

Propagation du flux de la droite vers la gauche

A	B	C	$\psi(C_1)$
0	0	0	0.0288
0	0	1	0.0216
0	1	0	0.0016
0	1	1	0.0072
1	0	0	0.0432
1	0	1	0.3024
1	1	0	0.016
1	1	1	0

C	$\psi(S_1)$
0	0.1036
1	0.2892

C	D	$\psi(C_2)$
0	0	0
0	1	0.1036
1	0	0
1	1	0.2892

D	$\psi(S_2)$
0	0
1	0.3928

D	E	$\psi(C_3)$
0	0	0
0	1	0
1	0	0.03928
1	1	0.35352

FIGURE B.5 – Rajout d'une évidence dans le réseau Bayésien R et circulation des flux

$P(A|D=1) : P(A|D) = \dfrac{0.0432+0.3024+0.016}{P(v)}$ avec $P(v) = 0.0288+0.0216+0.0016+0.0072+0.0432+0.3024+0.016 = 0.4208$ d'où $P(A|D) = 0.859315589$.

B.3 Conclusion

Nous avons vu brièvement dans cette section comment l'algorithme JLO est déployé en pratique. Cet algorithme est un algorithme d'inférence exacte, il existe aussi

pour les réseaux Bayésiens de grande taille des algorithmes d'inférence approchée. Il existe aussi des algorithmes symboliques [30, 132], c'est-à-dire des algorithmes qui fournissent in fine la formule correspondant à la probabilité recherchée, nous n'avons pas trouvé de bibliothèque C++ permettant ce genre de calcul, ce qui explique pourquoi nous n'avons pas essayé ce genre de méthode.

Publications de l'auteur

[1] Claude Aynaud, Coralie Bernay-Angeletti, Romuald Aufrère, Christophe Debain, and Roland Chapuis. Sélection des données et regard critique sur le résultat dans le cadre de la localisation de véhicules. In *GRETSI'2013*. Brest, France, September 2013.

[2] Claude Aynaud, Coralie Bernay-Angeletti, Roland Chapuis, Romuald Aufrère, and Christophe Debain. Vehicle localization by using a multi-modality top down approach. In *13 th International Conference on Control, Automation, Robotics and Vision*. 2014.

[3] Claude Aynaud, Coralie Bernay-Angeletti, Roland Chapuis, Romuald Aufrère, and Nadir Karam. Real time vehicle localization using a top-down process. In *17th International Conference on Information Fusion*. Salamanca, 2014.

[4] Claude Aynaud, Coralie Bernay-Angeletti, Roland Chapuis, Romuald Aufrère Aufrère, and Christophe Debain. Approche top-down de perception multisensorielle : application à la localisation de véhicules. *RFIA*, 2014.

[5] Coralie Bernay-Angeletti, Claude Aynaud, Roland Chapuis, and Romuald Aufrère. A top-down perception approach for object recognition. In *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2014.

Bibliographie

[6] http ://effistore.effidence.com/.

[7] https ://support.google.com/websearch/answer/166331.

[8] Nisar Ahmed et al. Towards cooperative bayesian human-robot perception : Theory, experiments, opportunities. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*. Washington, 2013.

[9] Mohannad Al-Khatib and Jean J. Saade. An efficient data-driven fuzzy approach to the motion planning problem of a mobile robot. *Fuzzy Sets and Systems*, 134(1) :65 – 82, 2003. ISSN 0165-0114. doi :http://dx.doi.org/10.1016/S0165-0114(02)00230-0. Fuzzy Set Techniques for Intelligent Robotic Systems.

[10] FaimyQ. Ansari, JitendraKumar Pal, Jainendra Shukla, G.C. Nandi, and Pavan Chakraborty. A cloud based robot localization technique. In Manish Parashar, Dinesh Kaushik, OmerF. Rana, Ravi Samtaney, Yuanyuan Yang, and Albert Zomaya, editors, *Contemporary Computing*, volume 306 of *Communications in Computer and Information Science*, pages 347–357. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-32128-3. doi :10.1007/978-3-642-32129-0_36.

[11] Edson Hiroshi Aoki, Arunabha Bagchi, Pranab Mandal, and Yvo Boers. A theoretical look at information-driven sensor management criteria. In *Proceedings of the 14th International Conference on Information Fusion (FUSION)*, pages 1–8. IEEE, 2011.

[12] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8) :966–1005, Aug 1988. ISSN 0018-9219. doi :10.1109/5.5968.

[13] Yaakov Bar-Shalom and Edison Tse. Tracking in a cluttered environment with probabilistic data association. *Automatica*, 11(5) :451 – 460, 1975. ISSN 0005-1098. doi :http://dx.doi.org/10.1016/0005-1098(75)90021-7.

[14] Bahram Behzadian, Pratik Agarwal, Wolfram Burgard, and Gian Diego Tipaldi. Monte carlo localization in hand-drawn maps. *arXiv preprint arXiv :1504.00522*, 2015.

[15] Kostas Bekris, Rahul Shome, Athanasios Krontiris, and Andrew Dobson. Cloud automation : Precomputing roadmaps for flexible manipulation. *IEEE Robotics & Automation Magazine : Special Issue on Emerging Advances and Applications in Automation, p. Under Review*, 2014.

[16] William J Beksi and Nikolaos Papanikolopoulos. Point cloud culling for robot vision tasks under communication constraints. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3747–3752. IEEE, 2014.

[17] Abder Rezak Benaskeur. Consistent fusion of correlated data sources. In *IEEE 28th Annual Conference of the Industrial Electronics Society*, volume 4, pages 2652–2656. IEEE, 2002.

[18] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2) :239–256, Feb 1992. ISSN 0162-8828. doi :10.1109/34.121791.

[19] Jeff Bilmes. On virtual evidence and soft evidence in bayesian networks. Technical Report UWEETR-2004-0016, University of Washington, Dept. of EE, 2004.

[20] Jose-Luis Blanco, Javier González-Jiménez, and Juan-Antonio Fernández-Madrigal. A robust, multi-hypothesis approach to matching occupancy grid maps. *Robotica*, 31 :687–701, 8 2013. ISSN 1469-8668. doi :10.1017/S0263574712000732.

[21] Nico Blodow, Lucian Cosmin Goron, Zoltan-Csaba Marton, Dejan Pangercic, Thomas Ruhr, Moritz Tenorth, and Michael Beetz. Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4263–4270. IEEE, 2011.

[22] Rolf Blumentritt and Ron Johnston. Towards a strategy for knowledge management. *Technology Analysis & Strategic Management*, 11(3) :287–300, 1999.

[23] Frédéric Bourgault, Aakash Chokshi, John Wang, Danelle Shah, Jonathan Schoenberg, Ramnath Iyer, Franco Cedano, and Mark Campbell. Scalable bayesian human-robot cooperation in mobile sensor networks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2342–2349. IEEE, 2008.

[24] Guillaume Bresson. *Localisation d'une flotte de véhicules communicants par approche de type SLAM visuel décentralisé*. Ph.D. thesis, Université Blaise Pascal-Clermont-Ferrand II, 2014.

[25] Bruce G Buchanan, Edward Hance Shortliffe, et al. *Rule-based expert systems*, volume 3. Addison-Wesley Reading, MA, 1984.

[26] Timothy J Buschman and Earl K Miller. Top-down versus bottom-up control of attention in the prefrontal and posterior parietal cortices. *science*, 315(5820) :1860–1862, 2007.

[27] Enrique Castillo. *Expert systems and probabilistic network models*. Springer Science & Business Media, 1997.

[28] Yiu-Tong Chan, Wing-Yue Tsui, Hing-Cheung So, and Pak-chung Ching. Time-of-arrival based localization under nlos conditions. *Vehicular Technology, IEEE Transactions on*, 55(1) :17–24, 2006.

[29] YT Chan, F Chan, W Read, BR Jackson, and BH Lee. Hybrid localization of an emitter by combining angle-of-arrival and received signal strength measurements. In *IEEE 27th Canadian Conference on Electrical and Computer Engineering*, pages 1–5. IEEE, 2014.

[30] Kuo-Chu Chang and Robert Fung. Symbolic probabilistic inference with continuous variables. In *Proceedings of the Seventh conference on Uncertainty in Artificial Intelligence*, pages 77–81. Morgan Kaufmann Publishers Inc., 1991.

[31] R.I. Chaplin, S. Gunetileke, and R.M. Hodgson. Initialising neural networks with a priori problem knowledge. In *Neural Networks for Signal Processing*

X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop, volume 1, pages 165–174 vol.1. 2000. ISSN 1089-3555. doi :10.1109/NNSP.2000.889407.

[32] Sung-Bae Cho. Neural-network classifiers for recognizing totally unconstrained handwritten numerals. *Neural Networks, IEEE Transactions on*, 8(1) :43–53, 1997.

[33] Kok Seng Chong and Lindsay Kleeman. Accurate odometry and error modelling for a mobile robot. In *Proceedings IEEE International Conference on Robotics and Automation*, volume 4, pages 2783–2788. IEEE, 1997.

[34] Elvira Consortium et al. Elvira : An environment for creating and using probabilistic graphical models. In *Proceedings of the first European workshop on probabilistic graphical models*, pages 222–230. 2002.

[35] Ingemar J Cox. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10(1) :53–66, 1993.

[36] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553) :503–507, 2015.

[37] Dina Aboul Dahab, Samy SA Ghoniemy, and Gamal M Selim. Automated brain tumor detection and identification using image processing and probabilistic neural network techniques. *International Journal of Image Processing and Visual Communication*, 1(2) :1–8, 2012.

[38] Randall Davis, Bruce G Buchanan, and Edward H Shortliffe. Retrospective on "production rules as a representation for a knowledge-based consultation program". *Artificial intelligence*, 59(1) :181–189, 1993.

[39] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam : Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6) :1052–1067, 2007.

[40] Joachim Denzler and Christopher M Brown. Information theoretic sensor data selection for active object recognition and state estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2) :145–157, 2002.

[41] Eric Driver and Darryl Morrell. Implementation of continuous bayesian networks using sums of weighted gaussians. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 134–140. Morgan Kaufmann Publishers Inc., 1995.

[42] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping : part i. *Robotics & Automation Magazine, IEEE*, 13(2) :99–110, 2006.

[43] Felix Duvallet and Ashley Desmond Tews. Wifi position estimation in industrial environments using gaussian processes. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2216–2221. IEEE, 2008.

[44] José Figueira, Salvatore Greco, and Matthias Ehrgott. *Multiple criteria decision analysis : state of the art surveys*, volume 78. Springer Science & Business Media, 2005.

[45] David Filliat and Jean-Arcady Meyer. Map-based navigation in mobile robots : : I. a review of localization strategies. *Cognitive Systems Research*, 4(4) :243–282, 2003.

[46] Georgios Floros, Benito van der Zander, and Bastian Leibe. Openstreetslam : Global vehicle localization using openstreetmaps. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1054–1059. IEEE, 2013.

[47] Richard Forsyth and Roy Rada. *Machine Learning : Applications in Expert Systems and Information Retrieval*. Halsted Press, New York, NY, USA, 1986. ISBN 0-470-20309-9.

[48] Satoshi Fujimoto, Zhencheng Hu, Claude Aynaud, and Roland Chapuis. Driving intention assistance for front-wheel-drive personal electric vehicle. *ppniv*, 2013.

[49] Dariu M Gavrila. Pedestrian detection from a moving vehicle. In *Computer Vision—ECCV 2000*, pages 37–49. Springer, 2000.

[50] Ken Goldberg and Ben Kehoe. Cloud robotics and automation : A survey of related work. Technical report, University of California, 2013.

[51] Joshua S Greenfeld. Matching gps observations to locations on a digital map. In *Transportation Research Board 81st Annual Meeting*. 2002.

[52] Hugo Grimmett, Rohan Paul, Rudolph Triebel, and Ingmar Posner. Knowing when we don't know : Introspective classification for mission-critical decision making. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4531–4538. IEEE, 2013.

[53] Dominique Gruyer, Alain Lambert, Mathias Perrollaz, and Denis Gingras. Experimental comparison of bayesian positioning methods based on multi–sensor data fusion. *International Journal of Vehicle Autonomous Systems*, 12(1) :24–43, 2014.

[54] Guofa Guo, Ning Wang, and Kaisheng Zhang. Distance difference algorithm based on rssi for highway vehicle localization. *Sensors & Transducers (1726-5479)*, 169, 2014.

[55] Fredrik Gustafsson and Fredrik Gunnarsson. Positioning using time-difference of arrival measurements. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 6, pages VI–553. IEEE, 2003.

[56] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d mapping : Using depth cameras for dense 3d modeling of indoor environments. In *Experimental robotics*, pages 477–491. Springer, 2014.

[57] Alfred O Hero and Douglas Cochran. Sensor management : Past, present, and future. *Sensors Journal, IEEE*, 11(12) :3064–3075, 2011.

[58] III Hero, AlfredO., ChristopherM. Kreucher, and Doron Blatt. Information theoretic approaches to sensor management. In III Hero, AlfredO., DavidA. Castañón, Douglas Cochran, and Keith Kastella, editors, *Foundations and Applications of Sensor Management*, pages 33–57. Springer US, 2008. ISBN 978-0-387-27892-6. doi :10.1007/978-0-387-49819-5_3.

[59] Albert S Huang, Stefanie Tellex, Abraham Bachrach, Thomas Kollar, Deb Roy, and Nicholas Roy. Natural language command of an autonomous micro-air vehicle. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2663–2669. IEEE, 2010.

[60] Cecil Huang and Adnan Darwiche. Inference in belief networks : A procedural guide. *International Journal of Approximate Reasoning*, 15(3) :225–263, 1996.

[61] David Janiszek and Damien Pellet. L'université paris descartes, 1ère université européenne, 2ème université mondiale, à acquérir un robot humanoïde, nao, dans le cadre d'un programme d'éducation, 2011.

[62] Finn V Jensen, Steffen L Lauritzen, and Kristian G Olesen. Bayesian updating in recursive graphical models by local computation. *Computational Statistics Quarterly*, 4(1) :269–282, 1990.

[63] J Jessup, SN Givigi, and A Beaulieu. Merging of octree based 3d occupancy grid maps. In *Systems Conference (SysCon), 2014 8th Annual IEEE*, pages 371–377. IEEE, 2014.

[64] J Jessup, SN Givigi, and A Beaulieu. Robust and efficient multi-robot 3d mapping with octree based occupancy grids. In *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*, pages 3996–4001. IEEE, 2014.

[65] Apoorva Jindal and Konstantinos Psounis. Modeling spatially correlated data in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 2(4) :466–499, 2006.

[66] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[67] Michael I Jordan et al. Why the logistic function ? a tutorial discussion on probabilities and neural networks, 1995.

[68] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

[69] Simon J Julier and Jeffrey K Uhlmann. General decentralized data fusion with covariance intersection (ci). 2001.

[70] Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. *Journal of Fluids Engineering*, 83(1) :95–108, 1961.

[71] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1) :35–45, 1960.

[72] Abraham Kandel. *Fuzzy expert systems*. CRC press, 1991.

[73] Tian Kangsheng and Zhu Guangxi. Sensor management based on fisher information gain. *Systems Engineering and Electronics, Journal of*, 17(3) :531–534, 2006.

[74] Evan Kaufman, Thomas Alan Lovell, and Taeyoung Lee. Optimal joint probabilistic data association filter avoiding coalescence in close proximity. In *Control Conference (ECC), 2014 European*, pages 2709–2714. IEEE, 2014.

[75] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg. A survey of research on cloud robotics and automation. *Automation Science and Engineering, IEEE Transactions on*, PP(99) :1–12, 2015. ISSN 1545-5955. doi :10.1109/TASE.2014.2376492.

[76] Davis E. King. Dlib-ml : A machine learning toolkit. *Journal of Machine Learning Research*, 10 :1755–1758, 2009.

[77] Uffe Kjærulff. Triangulation of graphs–algorithms giving small total state space. 1990.

[78] Sven Koenig, Apurva Mudgal, and Craig Tovey. A near-tight approximation lower bound and algorithm for the kidnapped robot problem. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 133–142. ACM, 2006.

[79] Mark P Kolba and Leslie M Collins. Information-based sensor management in the presence of uncertainty. *IEEE Transactions on Signal Processing*, 55(6) :2731–2735, 2007.

[80] David Kortenkamp and Terry Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *AAAI*, volume 94, pages 979–984. 1994.

[81] Ioannis Kostavelis and Antonios Gasteratos. Semantic mapping for mobile robotics tasks : A survey. *Robotics and Autonomous Systems*, 2014.

[82] J Krejsa and S Vechet. Infrared beacons based localization of mobile robot. *Elektronika ir Elektrotechnika*, 117(1) :17–22, 2012.

[83] Chris M Kreucher, Keith D Kastella, and Alfred O Hero III. Information-based sensor management for multitarget tracking. In *Optical Science and Technology, SPIE's 48th Annual Meeting*, pages 480–489. International Society for Optics and Photonics, 2003.

[84] Christopher M Kreucher, Alfred O Hero, Keith D Kastella, and Mark R Morelande. An information-based approach to sensor management in large dynamic networks. *Proceedings of the IEEE*, 95(5) :978–999, 2007.

[85] Benjamin Kuipers and Patrick Beeson. Bootstrap learning for place recognition. In *AAAI/IAAI*, pages 174–180. 2002.

[86] RYO KURAZUME and SHIGEO HIROSE. An experimental study of a cooperative positioning system. *Autonomous Robots*, 8 :43–52, 2000.

[87] Raphael Labayrade, Didier Aubert, and J-P Tarel. Real time obstacle detection in stereovision on non flat road geometry through" v-disparity" representation. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 646–651. IEEE, 2002.

[88] Laetitia Lamard, Roland Chapuis, and J-P Boyer. Dealing with occlusions with multi targets tracking algorithms for the real road context. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 371–376. IEEE, 2012.

[89] Pierre Lébraly. *Non-overlapping camera calibration and 3D reconstruction : Application to Vision-Based Robotics*. Theses, Université Blaise Pascal - Clermont-Ferrand II, January 2012.

[90] Tao Li, Mark G Petovello, Gerard Lachapelle, and Chaminda Basnayake. Ultra-tightly coupled gps/vehicle sensor integration for land vehicle navigation. *Navigation*, 57(4) :263–274, 2010.

[91] Dae-Woon Lim, Sung-Hoon Choi, and Joon-Suk Jun. Automated detection of all kinds of violations at a street intersection using real time individual vehicle tracking. In *Image Analysis and Interpretation, 2002. Proceedings. Fifth IEEE Southwest Symposium on*, pages 126–129. IEEE, 2002.

[92] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, 18(3) :249–275, 1997.

[93] Zheyuan Lu, Zhencheng Hu, and Keiichi Uchimura. Slam estimation in dynamic outdoor environments : A review. In Ming Xie, Youlun Xiong, Caihua Xiong, Honghai Liu, and Zhencheng Hu, editors, *Intelligent Robotics and Applications*, volume 5928 of *Lecture Notes in Computer Science*, pages 255–267. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-10816-7. doi : 10.1007/978-3-642-10817-4_25.

[94] D. Ma, M. Sandberg, and B. Jiang. Characterizing the heterogeneity of the openstreetmap data and community. *ArXiv e-prints*, March 2015.

[95] Raj Madhavan and Hugh F Durrant-Whyte. 2d map-building and localization in outdoor environments. *Journal of Robotic Systems*, 22(1) :45–63, 2005.

[96] Ronald Mahler. Global optimal sensor allocation. In *Proceedings of the Ninth National Symposium on Sensor Fusion*, volume 1, pages 167–172. 1996.

[97] Andras Majdik, Mircea Popa, Levente Tamas, Istvan Szoke, and Gheorghe Lazea. New approach in solving the kidnapped robot problem. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 1–6. June 2010.

[98] András L Majdik, Yves Albers-Schoenberg, and Davide Scaramuzza. Mav urban localization from google street view data. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3979–3986. IEEE, 2013.

[99] Florent Malartre and Pierre Delmas. http ://www.4d-virtualiz.com/index.html, 2011.

[100] Jean-Arcady Meyer and David Filliat. Map-based navigation in mobile robots : : Ii. a review of map-learning and path-planning strategies. *Cognitive Systems Research*, 4(4) :283 – 317, 2003. ISSN 1389-0417. doi :http://dx.doi.org/10.1016/S1389-0417(03)00007-X.

[101] G. Mohanarajah, D. Hunziker, R. D'Andrea, and M. Waibel. Rapyuta : A cloud robotics platform. *Automation Science and Engineering, IEEE Transactions on*, 12(2) :481–493, April 2015. ISSN 1545-5955. doi :10.1109/TASE.2014.2329556.

[102] Karim EL Mokhtari. *Estimation circulaire multi-modèles appliquée au Map matching en environnement contraint*. Ph.D. thesis, Université Abdelmalek Essaâdi, Faculté des Sciences et Techniques, Tanger - Maroc, 2015.

[103] Julien Moras. *Grilles de perception évidentielles pour la navigation robotique en milieu urbain*. Ph.D. thesis, Université de Technologie de Compiègne, 2013.

[104] Oscar Martinez Mozos, Rudolph Triebel, Patric Jensfelt, Axel Rottmann, and Wolfram Burgard. Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*, 55(5) :391–402, 2007.

[105] Kevin P Murphy. A variational approximation for bayesian networks with discrete and continuous latent variables. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 457–466. Morgan Kaufmann Publishers Inc., 1999.

[106] Patrick Naïm, Pierre-Henri Wuillemin, Philippe Leray, Olivier Pourret, and Anna Becker. *Réseaux bayésiens*. Editions Eyrolles, 2011.

[107] José Neira and Juan D Tardós. Data association in stochastic mapping using the joint compatibility test. *Robotics and Automation, IEEE Transactions on*, 17(6) :890–897, 2001.

[108] Pascal Neis, Dennis Zielstra, and Alexander Zipf. The street network evolution of crowdsourced maps : Openstreetmap in germany 2007–2011. *Future Internet*, 4(1) :1–21, 2011.

[109] Maria Niessen, Gert Kootstra, Sjoerd De Jong, and Tjeerd Andringa. Expectancy-based robot localization through context evaluation. In *Proceedings of the International Conference on Artificial Intelligence*, pages 371–377. CSREA Press, 2009.

[110] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004.

[111] Aboelmagd Noureldin, Ahmed El-Shafie, and Mohamed Bayoumi. Gps/ins integration utilizing dynamic neural networks for vehicular navigation. *Information Fusion*, 12(1) :48 – 57, 2011. ISSN 1566-2535. doi :http://dx.doi.org/10.1016/j.inffus.2010.01.003. Special Issue on Intelligent Transportation Systems.

[112] Andreas Nüchter and Joachim Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11) :915–926, 2008.

[113] Edwin B Olson. Real-time correlative scan matching. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4387–4393. IEEE, 2009.

[114] Judea Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann, 1988.

[115] Rong Peng and Mihail L Sichitiu. Angle of arrival localization for wireless sensor networks. In *Sensor and Ad Hoc Communications and Networks, 2006. SECON'06. 2006 3rd Annual IEEE Communications Society on*, volume 1, pages 374–382. IEEE, 2006.

[116] Giovanni Pezzulo. Coordinating with the future : The anticipatory nature of representation. *Minds and Machines*, 18(2) :179–225, June 2008. ISSN 0924-6495, 1572-8641. doi :10.1007/s11023-008-9095-5.

[117] Giovanni Pezzulo, Gianluca Baldassarre, Martin V Butz, Cristiano Castelfranchi, and Joachim Hoffmann. From actions to goals and vice-versa : Theoretical analysis and models of the ideomotor principle and tote. In *Anticipatory Behavior in Adaptive Learning Systems*, pages 73–93. Springer, 2007.

[118] S.S. Prabha, A.J.P. Antony, M.J. Meena, and S.R. Pandian. Smart cloud robot using raspberry pi. In *Recent Trends in Information Technology (ICRTIT), 2014 International Conference on*, pages 1–5. April 2014. doi :10.1109/ICRTIT.2014.6996193.

[119] Mohammed A Quddus. *High integrity map matching algorithms for advanced transport telematics applications*. Ph.D. thesis, Imperial College London, United Kingdom, 2006.

[120] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1) :81–106, 1986.

[121] Janusz Racz and Artur Dubrawski. Artificial neural network for mobile robot topological localization. *Robotics and autonomous systems*, 16(1) :73–80, 1995.

[122] AM Ramiya, Rama Rao Nidamanuri, and R Krishnan. Semantic labelling of urban point cloud data. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1 :907–911, 2014.

[123] Paul Ramsey et al. Postgis manual. *Refractions Research Inc*, 2005.

[124] Ziad Sankari and Hojjat Adeli. Probabilistic neural networks for diagnosis of alzheimer's disease using conventional and wavelet coherence. *Journal of neuroscience methods*, 197(1) :165–170, 2011.

[125] Martin Sarter, Ben Givens, and John P Bruno. The cognitive neuroscience of sustained attention : where top-down meets bottom-up. *Brain research reviews*, 35(2) :146–160, 2001.

[126] Basic Transmission Scheme. Lte : the evolution of mobile broadband. *IEEE Communications magazine*, page 45, 2009.

[127] Bernt Schiele and James L Crowley. A comparison of position estimation techniques using occupancy grids. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 1628–1634. IEEE, 1994.

[128] A. Schindler, G. Maier, and F. Janda. Generation of high precision digital maps using circular arc splines. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 246–251. June 2012. ISSN 1931-0587. doi :10.1109/IVS.2012.6232124.

[129] A. Schlichting and C. Brenner. Localization using automotive laser scanners and local pattern matching. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pages 414–419. June 2014. doi :10.1109/IVS.2014.6856460.

[130] Matthias Roland Schmid, Mirko Maehlisch, Jürgen Dickmann, and H-J Wuensche. Dynamic level of detail 3d occupancy grids for automotive use. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 269–274. IEEE, 2010.

[131] Brent Schwarz. Lidar : Mapping the world in 3d. *Nature Photonics*, 4(7) :429–430, 2010.

[132] Ross D Shachter, Bruce D'Ambrosio, and Brendan Del Favero. Symbolic probabilistic inference in belief networks. In *AAAI*, volume 90, pages 126–131. 1990.

[133] Danelle Shah, Joseph Schneider, and Mark Campbell. A sketch interface for robust and natural robot control. *Proceedings of the IEEE*, 100(3) :604–622, 2012.

[134] JR Siddiqui and Siamak Khatibi. Semantic urban maps. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 4050–4055. IEEE, 2014.

[135] Cherif Smaili. *Fusion de données multi-capteurs à l'aide d'un réseau bayésien pour l'estimation d'état d'un véhicule*. Ph.D. thesis, Université Nancy II, 2010.

[136] Jason Stauch, M Jah, J Baldwin, Tom Kelecy, and Keric Hill. Mutual application of joint probabilistic data association, filtering, and smoothing techniques for robust multiple space object tracking. In *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, no. AIAA-2014-4365*. 2014.

[137] Hanne Stensola, Tor Stensola, Trygve Solstad, Kristian Frøland, May-Britt Moser, and Edvard I Moser. The entorhinal grid map is discretized. *Nature*, 492(7427) :72–78, 2012.

[138] Zui Tao and Philippe Bonnifait. Tightly coupling gps with lane markings for autonomous vehicle navigation. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 439–444. IEEE, 2014.

[139] Robert E Tarjan and Mihalis Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on computing*, 13(3) :566–579, 1984.

[140] Stefanie Tellex, Pratiksha Thakerll, Robin Deitsl, Dimitar Simeonovl, Thomas Kollar, and Nicholas Royl. Toward information theoretic human-robot dialog. *Robotics*, page 409, 2013.

[141] Cédric Tessier, Christophe Cariou, Christophe Debain, Frederic Chausse, Roland Chapuis, and Christophe Rousset. A real-time, multi-sensor architecture for fusion of delayed observations : application to vehicle localization. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*, pages 1316–1321. IEEE, 2006.

[142] Cédric TESSIER. *Système de localisation basé sur une stratégie de perception cognitive appliqué à la navigation autonome d'un robot mobile*. Ph.D. thesis, Université Blaise Pascal Clermont II, Novembre 2007.

[143] Cédric Tessier, Christophe Debain, Roland Chapuis, and Frédéric Chausse. Map aided localization and vehicle guidance using an active landmark search. *Information Fusion*, Volume 11, Issue 3 :283–296, 2010.

[144] Sebastian Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous robots*, 15(2) :111–127, 2003.

[145] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

[146] Edward C Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4) :189, 1948.

[147] Jurgen Vandorpe, Hendrik Van Brussel, and Hong Xu. Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 901–908. IEEE, 1996.

[148] Shrihari Vasudevan, Stefan Gächter, Viet Nguyen, and Roland Siegwart. Cognitive maps for mobile robots—an object based approach. *Robotics and Autonomous Systems*, 55(5) :359–371, 2007.

[149] Javier Velez, Garrett Hemann, Albert S Huang, Ingmar Posner, and Nicholas Roy. Active exploration for robust object detection. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2752–2757. AAAI Press, 2011.

[150] J. Vilca, L. Adouane, and Y. Mezouar. Reactive navigation of a mobile robot using elliptic trajectories and effective online obstacle detection. *Gyroscopy and Navigation*, 4(1) :14–25, 2013. ISSN 2075-1087. doi :10.1134/S2075108713010094.

[151] Damien Vivet. *Perception of the environment with a hyper-frequency radar. Application to simultaneous localization and mapping, to detection and tracking of moving objects in outdoor environment.* Theses, Université Blaise Pascal - Clermont-Ferrand II, December 2011.

[152] Damien Vivet, Franck Gérossier, Paul Checchin, Laurent Trassoudaine, and Roland Chapuis. Mobile ground-based radar sensor for localization and mapping : An evaluation of two approaches. *International Journal of Advanced Robotic Systems*, 10(307) :12, 2013.

[153] Christopher E White, David Bernstein, and Alain L Kornhauser. Some map matching algorithms for personal navigation assistants. *Transportation Research Part C : Emerging Technologies*, 8(1) :91–108, 2000.

[154] Ning Xiong and Per Svensson. Multi-sensor management for information fusion : issues and approaches. *Information fusion*, 3(2) :163–186, 2002.

[155] Guoxian Zhang, Silvia Ferrari, and M Qian. An information roadmap method for robotic sensor path planning. *Journal of Intelligent and Robotic Systems*, 56(1-2) :69–98, 2009.

[156] Yuanliang Zhang and KilTo Chong. An gps/dr navigation system using neural network for mobile robot. *International Journal of Precision Engineering and Manufacturing*, 15(12) :2513–2519, 2014. ISSN 2234-7593. doi : 10.1007/s12541-014-0622-4.

[157] Liang Zhao and Charles E Thorpe. Stereo-and neural network-based pedestrian detection. *Intelligent Transportation Systems, IEEE Transactions on*, 1(3) :148–154, 2000.